2009 IEEE International Enterprise Distributed Object Computing Conference

Modelling and Reasoning Languages for Social Networks Policies

Guido Governatori and Renato Iannella
NICTA, Queensland Research Laboratory,
St Lucia, QLD, 4072, Australia
{guido.governatori,renato}@nicta.com.au

Abstract—**Policy languages (such as privacy and rights) have had little impact on the wider community. Now that Social Networks have taken off, the need to revisit Policy languages and realign them towards Social Networks requirements has become more apparent. One such language is explored as to its applicability to the Social Networks masses. We also argue that policy languages alone are not sufficient and thus they should be paired with reasoning mechanisms to provide precise and unambiguous execution models of the policies. To this end we propose a computationally oriented model to represent, reason with and execute policies for Social Networks.**

Index Terms—**Social Networks; Open Digital Rights Language (ODRL); Policy; Privacy; Rights; Defeasible Logic;**

I. Introduction

The Web undoubtedly has developed an impressive collection of technologies for supporting sophisticated information sharing and representation. Social Networks – via the innovative use of Web 2.0 features – have also taken the wider web community by surprise with such rapid uptake and widespread sharing of user-generated content.

A major lesson from Social Networks is that by offering "simplicity and efficiency" we can attract mass audiences [?]. Equally, a major lesson from the Semantic Web is that we can "deliver information directly to people for whom the information was relevant" by adopting a semantically-aware social networking stack across Social Network services [?]. The challenge now is to bring these two communities together in such a way that the technology (e.g., Semantic Web) meets the needs of a mass audience (e.g., Social Networks).

Social Networks have highlighted one particular area of concern:

> "They provide complex and indeterminate mechanisms to specific privacy and other policies for protecting access to personal information, and allow information to be shared that typically would not follow social and professional norms." [?]

There have been numerous attempts to solve this problem in the past [?] but none have been really successful, nor applicable to the Social Networks community. A new approach is required to manage seamless policy interaction for the Social Networks masses.

This raises four key challenges for policy languages:

- Policy Expression – how to unambiguously define the terms and conditions of a policy.
- Policy Transparency – how to ensure all parties are aware of the policy and its implications.
- Policy Conflict – how to detect potential incompatibilities between dependent policies.
- Policy Accountability – how to track policy exceptions and obligations.

With the emergence of Social Network interactions, the four policy challenges now need to be aligned with this new environment. Traditionally policy languages were designed based on a transaction environment. That is, the content and parties would enter into some explicit agreement under the control of some policy management system, for example, a DRM system buying music. However the Social Networks user base is more inclined to share content with friends and colleagues without any predetermined agreement. The focus has moved away from the transaction-based constrained policy (e.g., play the video 5 times over a 2 month period) to a regime based on sharing content to dynamic groups of people (e.g., any friend can comment on these photos.)

In this paper we first look at the Policy Expression Challenge through the emerging development of the Open Digital Rights Language (ODRL). We then look at a specific Use Case from Social Networks and apply the ODRL policy language to validate its expressiveness. We also investigate a computationally oriented approach for the formalisation and execution of policies. We then show the mapping between the formal model and ORDL. We conclude with a summary of the future research work and potential directions for policy challenges for Social Networks.

II. ODRL Version 2.0

The Open Digital Rights Language (ODRL) Version 2.0 Model [?] has evolved over the past years from a specific rights management language to a more generic policy language. Figure 1 shows the core entities of the underlying information model. Like many other policy languages at the time, ODRL was modeled on the transaction-based policy environment. The new version of ODRL is motivated to broaden this scope to incorporate the unique needs of Social Network policies.

The key ideas of the ODRL model that are applicable to Social Networks include:

- A clear identification of the Asset (for any type of Social Network content).

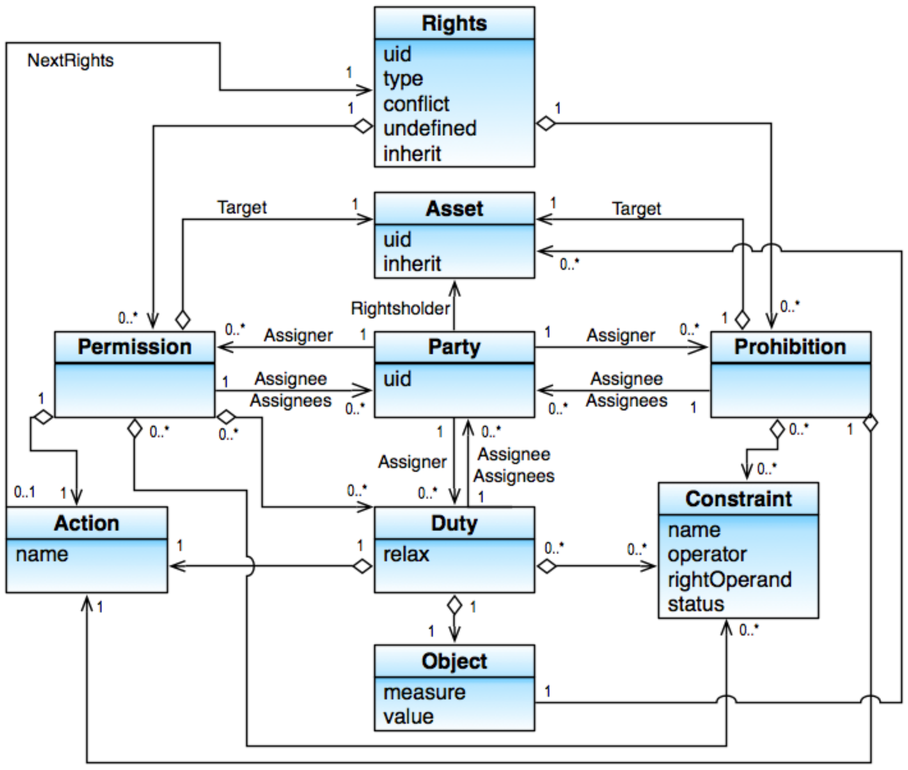


Fig. 1. ODRL Version 2.0 Model

- Actions that are allowed to be performed (Permissions) or not allowed to be performed (Prohibitions) can be articulated.
- All the Parties involved can be specified (who assigns rights to whom).
- Any Duties on Parties can be stipulated (their obligations that must be meet).
- Constraints can be enumerated for any of the key entities.

The ODRL Version 2.0 Model is defined abstractly in UML. This was undertaken to ensure that the semantics of the policy language could standalone and was not dependent on any other underlying encoding model or syntax. Additionally, it focusses the work on Policy semantics and does not try and force the semantics into another framework.

The ODRL Version 2.0 Model will be implementable via both XML Schema and Semantic Web specifications but is still under development. However, it has reached the stage where it can be applied to different scenarios to validate its applicability to various communities.

III. Social Networks Requirements

We looked at two popular Social Networks (FaceBook and Flickr) and reviewed the types of conditions (or constraints) that can be applied to their content. Figure 2 shows examples of these conditions from these two Social Networks. The findings were also consistent with Professional Networks, such as LinkedIn and Plaxo.

What is clear from Figure 2 is that the policy decision points are focussed on constraining who the end user party is. That is, the content owner can specify these general types of limitations for who can access their content:

- Only the content owner (i.e., no one else)
- Specific (named) friends and colleagues (both allowed and not allowed)
- All direct friends or colleagues
- Your second level friends or colleagues (i.e., friends of friends)
- All Groups (that the content owner is a member of) or some Groups
- Everyone (i.e., public)

IV. Social Networks Use Case

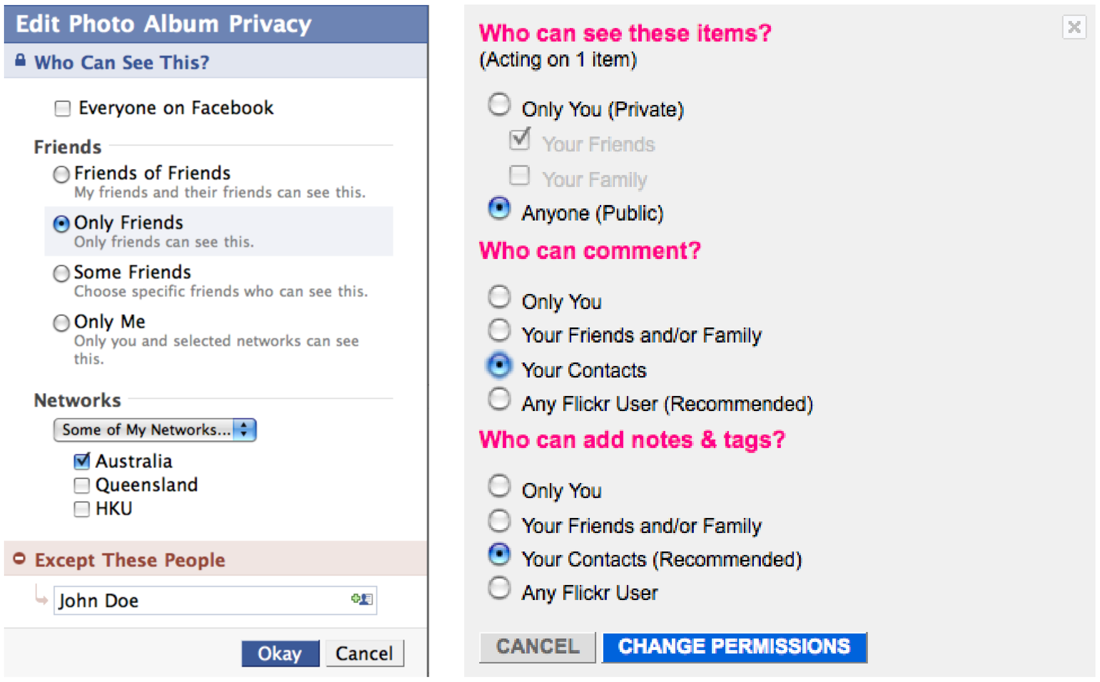
We looked at a specific use case [?] from the W3C Future of Social Networks Workshop (Jan 2009) in which:

> "Alice wants to give access to her wedding pictures only to people that are fellows on both Flickr and Twitter and that have a blog she commented at least twice during the last 10 days."

This use case also follows the similar theme in which the rights are bestowed on a constrained group of people. In this case, the group has additional constraints and requirements, specifically that Alice has commented on their Blog in the past 10 days. We then looked at expressing this use case in ODRL Version 2.0.

V. ODRL 2.0 Meets Web 2.0 (aka Social Networks)

Deconstructing the Alice Use Case leads to the following language expression needs:



Edit Photo Album Privacy

Who Can See This?

☐ Everyone on Facebook

Friends

Friends of Friends
My friends and their friends can see this.

[✓ Only Friends
Only friends can see this.]

Some Friends
Choose specific friends who can see this.

Only Me
Only you and selected networks can see this.

Networks

Some of My Networks...

[✓ Australia]
☐ Queensland
☐ HKU

Except These People

John Doe

Who can see these items?
(Acting on 1 item)

Only You (Private)
[✓ Your Friends]
☐ Your Family
[✓ Anyone (Public)]

Who can comment?

Only You
Your Friends and/or Family
[✓ Your Contacts]
Any Flickr User (Recommended)

Who can add notes & tags?

Only You
Your Friends and/or Family
[✓ Your Contacts (Recommended)]
Any Flickr User

Fig. 2. FaceBook (left) and Flickr (right) Privacy Settings

- Identifying the Wedding Photos
- Alice is assigning rights
- The permission is viewing
- The recipient of the permission is the group of people that meet all of these criteria
  - Members of Flickr and Twitter, and
  - Have Blog sites, and
  - Alice has commented at least twice on these blogs,
  - In the last 10 days.

Figure 3 shows this instance of the Alice Use Case expressed in the ODRL Version 2.0 Model.

The instance shows the Wedding Photos Asset being the subject of a View Permission assigned by the Party Alice. The Assignee of the policy is the generic ODRL "anyone" Party that has three specific Duties that must be meet. These Duties then filter the "anyone" Party to the specific parties that match Alice's use case. In this case they must be members of Flickr and Twitter and have a Blog. The latter then has an additional constraint that Alice has commented on their blog in the past 10 days.

This mapping exercise of the Use Case has highlighted some additional requirements for the ODRL Version 2.0 evolution. Specifically, it has identified the need for a generic "anyone" party and membership semantics that need to be part of the ODRL Core Metadata (currently in Working Draft status).

VI. ODRL 2.0 Meets Web 3.0 (aka Semantic Web)

ODRL Version 1.1 has been widely deployed primarily in XML (with over a billion mobile handsets supporting the OMA profile of the ODRL language). ODRL Version 2.0 will also support an RDF/XML binding to capitalize on the Semantic Web opportunities. The binding will require some hard decisions on how to best fit the ODRL model into the RDF Model. For example, below is a potential RDF/XML encoding of the Blog Duty:

```
<odrl:Duty>
  <odrl:action rdf:resource="odrl:hasA"/>
  <odrl:object rdf:parseType="Resource">
    <rdf:value rdf:resource="odrl:TRUE"/>
    <odrl:measure rdf:resource="urn:Blog"/>
  </odrl:object>
  <odrl:container rdf:parseType="Collection">
    <odrl:constraint rdf:about="urn:odrl:AND"/>
    <odrl:constraint rdf:ID="constraint-01"/>
    <odrl:constraint rdf:ID="constraint-02"/>
    <odrl:constraint rdf:ID="constraint-03"/>
 </odrl:container>
</odrl:Duty>
```

Some encoding issues will need to be addressed, such as how to best fit the ODRL Container model into the RDF model. For example, should the RDF Collection include the ODRL Boolean operator as a member or are there better ways that RDF supports this.

A major issue will be the mapping of the ODRL Permission/Prohibition model into RDF to infer policy conflicts. ODRL Version 2.0 has introduced a precedence mechanism to guide conflict detection. How this can be best modelled in RDF will be an interesting challenge in itself. We can imagine an extension to the Alice use case where she has also given permission for anyone who is a member of MySpace full access to all her photos. This will be in conflict with her original use case as now "all her photos" includes her wedding photos. How can we detect this and warn her of this conflict?

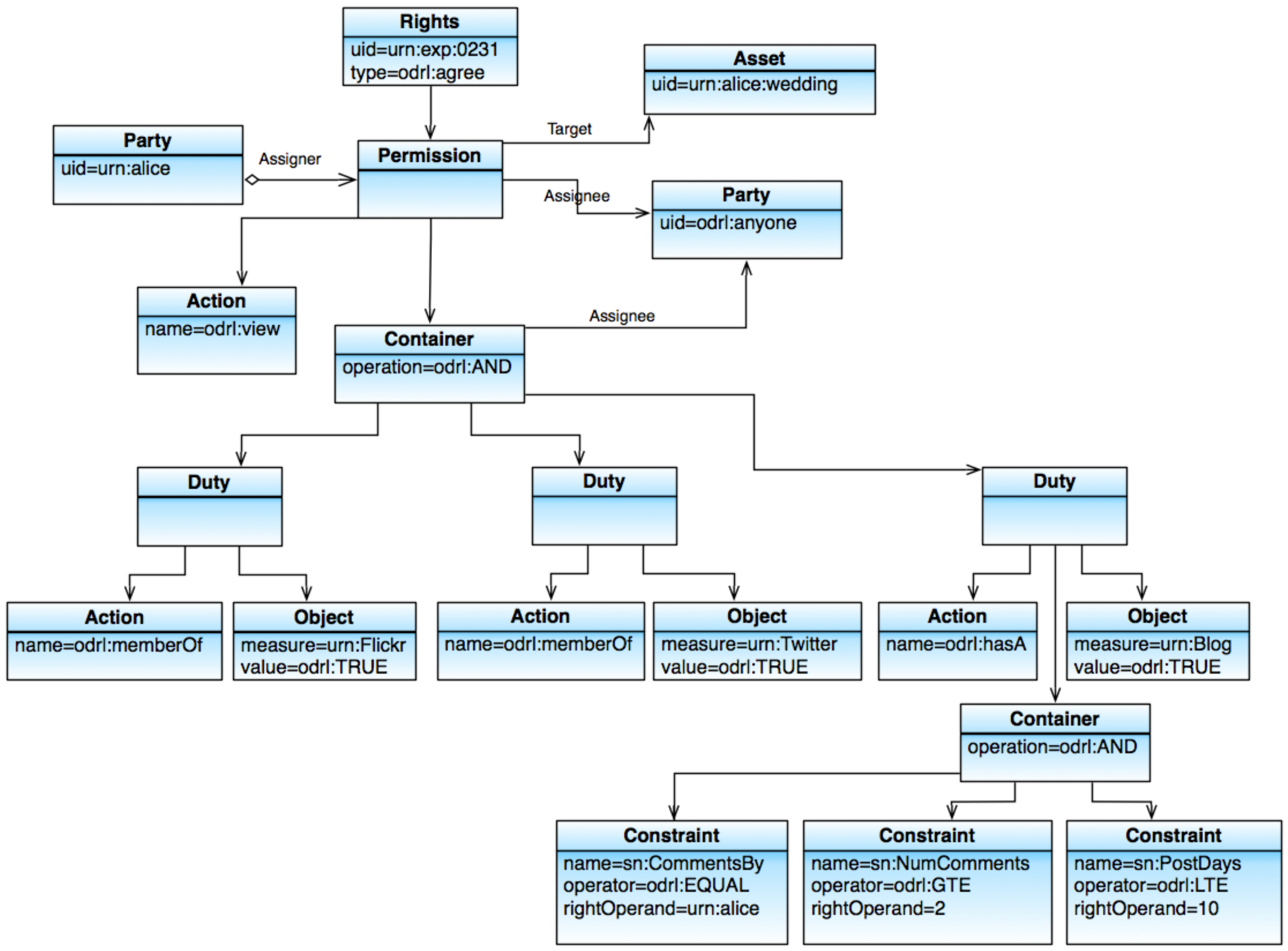


Fig. 3. Alice Use Case in ODRL Version 2.0

VII. Computing Policies

In the previous sections we have examined some essential requirements for a policy language for social networks, and we have seen to what extent ODRL meets those requirements. However, a policy language has to be implemented to identify the properties enjoyed by resources and members in a network. In this section we provide a computationally oriented approach to this problem. In addition we examine some further aspects relevant to the deployment of executable specifications for policies in social networks.

Our proposal to address the issue of how to implement a policy language is based on FCL, a logical approach proposed by Governatori [?] for the representation of an executable contract language and further proposed for the study of compliance of business processes [?].

To illustrate some of the features needed we extend the Alice example.

Suppose that the network offers members the facility to create blacklists where a member can list members of the networks that cannot access the member resources, and the user can specify restrictions on the resources available to members in a blacklist. Alice decided that blacklisted members cannot access her resources at all. Moreover, suppose that Alice put Bob in the photo blacklist, but she has posted a few recent comments on Bob's blog, and Bob is a member of the categories listed in Alice conditions to access her wedding pictures.

Consider another example: The network has another feature. Each user has a profile page, and the user has to upload a picture to the profile page, and this picture is available to everybody in the network. Members who do not comply with the above conditions cannot access other members' private resources.

Alice puts a picture of her wedding as her public photo. Carl is another Flickr and Twitter fellow of Alice (not in her blacklist, and she repeatedly posted in his blog during the past week) who does not have his public picture in his profile.

The above examples illustrated some important features we are faced with when we want to implement a policy language in social networks (and not only in social networks).

- policy conditions have a normative nature;

- policy conditions can have exceptions;
- conditions in policy can conflict with each other;
- policies in a social networks can come from different sources;
- policy conditions sometimes involve violations of other policy conditions.

A. Executable Policy Specifications

We briefly present the basic of FCL (Formal Contract Logic). FCL results from the combination of an efficient rule base non-monotonic logic formalism (Defeasible Logic [?], [?]) and a Deontic Logic of violations [?]. Deontic Logic is the branch of logic studying normative like concepts like obligations, permissions, prohibitions, ... These notions correspond to the right, duty, prohibition notions illustrated in the previous sections. Deontic logic extends first order logic with the deontic operators O, P and F denoting obligations, permissions and prohibitions. The deontic operators satisfy the following equivalence relations:

$$OA \equiv \neg P \neg A \quad \neg O \neg A \equiv PA \quad O \neg A \equiv FA \quad \neg PA \equiv FA.$$

The operators also satisfy the following relationship $OA \rightarrow PA$, meaning that if A is obligatory, then A is permitted. This relationship can be used to ensure checking of the internal consistency of the obligations in a set of norms, i.e., whether it is possible to execute obligations without doing something that is forbidden. FCL then extend deontic logic by considering directed deontic operators. This means that each operator can be indexed by the subject and the beneficiary of the normative concept. Thus for example $O_s^b A$ means that s has the obligation of B with respect to b, where A could be the statement "prevent disclosure of personal information". Similarly $F_s B$ means that B is forbidden for s (where B, for example, means "access private data"). The deontic logic component of FCL gives us the ability to handle the normative aspects of ODRL, as well as the aspect of how to handle violations. Often the treatment of violations is not properly addressed in other deontic logics (see [?] for a detailed presentation of the problems related to violations in deontic logic).

Typically normative systems, of which policies are particular instances, include conditions that are activated to compensate breaches of other conditions. To capture this aspect we introduce the *reparation* (or compensation) operator $\otimes$, to be used in expression like $OA \otimes OB$. The meaning of an expression like $OA \otimes OB$ is that we have the obligation of A (i.e., OA), but in case this is violated, i.e., we have the negation of A, i.e., $\neg A$, then the obligation OB is in force. This means that achieving B compensate for failing to fulfill the obligation OA. Since the compensation is an obligation as well, it is possible that it is violated, and so it can trigger an additional compensation. To accommodate this we allow chains of obligation compensation of any length. Thus, for example we can have expressions like

$$OA_1 \otimes \cdots \otimes OA_n$$

(also called obligations chains or simply chains) saying that the main obligation is OA_1 but in case this is violated, then the next obligation is OA_2, and even if this is violated, then we trigger the obligation OA_3 and so on. In these chains we can have obligations and prohibitions[1]. Permissions can appear only as the last element of a chain. The reason for this is that it is not possible to have a violation of a permission, and so it is meaningless to have a compensation for something that cannot be violated.

The defeasible logic component of FCL allows us to capture exceptions and conflicts, more specifically the inference mechanism of FCL is an extension of Defeasible Logic.

Defeasible logic, originally created by Donald Nute [?] with a particular concern about efficiency and implementation, is a simple and efficient rule based non-monotonic formalism. Over the years, the logic has been developed and extended, and several variants have been proposed to model different aspects of normative reasoning and it encompasses other formalisms for normative reasoning.

The main intuition of the logic is to be able to derive "plausible" conclusions from partial and sometimes conflicting information. Conclusions are *tentative* conclusions in the sense that a conclusion can be withdrawn when we have new pieces of information.[2]

The knowledge in a Defeasible Theory is organised in *facts* and *rules* and *superiority relation*.

- Facts are indisputable statements.
- Defeasible rules are rules that can be defeated by contrary evidence.
- The superiority relation is a binary relation defined over the set of rules. The superiority relation determines the relative strength of two (conflicting) rules.

The meaning of a defeasible rule, like

$$A_1, \ldots, A_n \Rightarrow C$$

is that normally we are allowed to derive C given $A_1, \ldots, A_n$, unless we have some reasons to support the opposite conclusion (i.e., we have a rule like $B_1, \ldots, B_m \Rightarrow \neg C$).

Defeasible Logic is a "skeptical" non-monotonic logic, meaning that it does not support contradictory conclusions. Instead, Defeasible Logic seeks to resolve conflicts. In cases where there is some support for concluding A but also support for concluding $\neg A$, Defeasible Logic does not conclude either of them (thus the name skeptical). If the support for A has priority over the support for $\neg A$ then A is concluded.

A defeasible conclusion is a tentative conclusion that might be withdrawn by new pieces of information, or in other terms it is the 'best' conclusion we can reach with the given information. In addition, the logic is able to tell whether a conclusion is or is not provable. Thus, it is possible to have the following two types of conclusions:

[1]Please remember that prohibitions can be represented as obligations, $FA \equiv O\neg A$.

[2]For a full presentation of the logic, refer to [?], [?]

- Positive defeasible conclusions: meaning that the conclusions can be defeasible proved;
- Negative defeasible conclusions: meaning that one can show that the conclusion is not even defeasibly provable.

A (positive) defeasible conclusion A can be derived if there is a rule whose conclusion is A, whose prerequisites (antecedent) have either already been proved or given in the case at hand (i.e., facts), and any stronger rule whose conclusion is $\neg A$ (the negation of A) has prerequisites that fail to be derived. In other words, a conclusion A is (defeasibly) derivable when:

1) A is a fact; or
2) there is an applicable defeasible rule for A, and either
    a) all the rules for $\neg A$ are discarded (i.e., not applicable) or
    b) every applicable rule for $\neg A$ is weaker than an applicable strict or defeasible rule for A.

A rule is applicable if all elements in the body of the rule are derivable (i.e., all the premises are positively provable), and a rule is discarded if at least one of the elements of the body is not provable (or it is a negative defeasible conclusion).

The main difference between Defeasible logic and FCL is that in FCL the conclusion of a rule is an obligation chain (possibly a trivial chain where we have only one element).

Accordingly the reasoning mechanism to derive conclusion is an extension of that for defeasible logic. In defeasible logic the conclusion of a rule is a single literal and not a reparation chain. Thus, the condition that OA appears in the conclusion of a rule means in defeasible logic that OA is the conclusions of the rule. FCL extends defeasible logic with reparation chains, thus, we have to extend the reasoning mechanism of defeasible logic to accommodate the additional construction provided by FCL. To prove OA, we have to consider all rules with a reparation chain for OA, where for all elements before OA in the chain, the negation of the element is already provable. Thus to prove A given a rule

$$P_1,\dots,P_n \Rightarrow OC_1 \otimes \cdots \otimes OC_m \otimes OA \otimes OD_1 \otimes \cdots \otimes OD_k,$$

we have that $P_1,\dots,P_n$ must be all provable, and so must be $\neg C_1,\dots,\neg C_m$. For the full details see [**?**].

B. FCL at Work for Social Networks

In this section we will examine how the feature of FCL presented above address some important aspects of social network policies.

1) Exceptions: The superiority relation can be used to model exceptions. An exception is a situation where we have some specific information that prevents the derivation of the otherwise normal conclusion. Consider the following policy for handling accessing resources in a social network: 'Member resources on the network can be access by everybody, unless a resource is declared private'.

This condition can be expressed by the default rule

$$r_1 : \mathit{resource}(x) \Rightarrow P\,\mathit{access}(x)$$

The predicate *resource* states that x is a resource in the network, and then, the rule states that if something is a resource then access is permitted to it. In this case we can use an undirected permission to represent the condition (everybody in the network can access the resource, thus there is no need to specify a specific beneficiary for it). To capture the *unless* part of the policy, expressing the exception, we can use the rule

$$r_2 : \mathit{private}(x), \neg \mathit{owner}(x,y) \Rightarrow O_y \neg \mathit{access}(x)$$

The meaning of this rule is that if something is a private resource and somebody (y) is not the owner of the resources, then it is forbidden to y to access the resource. Remember that a prohibition is an obligation followed by a negation, i.e., $O\neg$.

In case both rules apply, then we cannot conclude anything, because the two rules are in conflict, something is at the same time permitted and prohibited. Thus to express that we have an exception we have to specify the superiority relation between the two rules. In this case, we can specify that $r_1 \prec r_2$, making thus r_2 stronger than r_1. This means that in case both fire, r_2 takes precedence over r_1, and we can derive the conclusion of r_1. Accordingly we can deny access to private resources.

2) Conflicts: In a social network we can have policies from multiple sources, for example we can have multiple policies from networks in a social network aggregator as well as policies from individual members. As result it is possible that policies are in conflict with other policies.

Let us go back to the Alice scenario. Her rule about access to her wedding pictures can be formalised as follows;

$$\begin{aligned} p_1 : &\mathit{wedding\_photo}(x), \mathit{flickr}(y), \mathit{twitter}(y),\\ &\mathit{blog}(z,y), \mathit{posted}(a,z,t_1), \mathit{posted}(a,z,t_2),\\ &t_1 > \mathit{Now}-10, t_2 > \mathit{Now}-10 \Rightarrow P_y \mathit{access}(x) \qquad (1)\end{aligned}$$

The predicates in the rule mean, respectively: x is the id of a wedding photo, y is a friend of Alice in Flickr, y is a friend of Alice in Twitter, z is the blog of y, Alice posted to the z's blog at time t_1, and at time t_2, and t_1 and t_2 are less than ten days ago (we assume that *Now* returns the current time in days). Then the rule says that if all the conditions above are satisfied then y is permitted to access the picture whose id is x.

Let us suppose now that x is a private wedding picture and that the other conditions in the wedding rule p_1 are satisfied. Given the two rules we have a possible conflict between the two rules. According to rule p_1 access should be granted. On the other hand, the restriction on private resources, rule r_2, prevents granting the access to the picture.

Again to solve the issue we have to use the superiority relation. Here one has to set that $r_1 \prec p_1$.

Notice that conflicts can arise both from rules from different sources, but, as in the case of exceptions, they can originate from one and the same source, as the following rule illustrate.

Consider the policy about blacklisting 'if a member is a in blacklist then the member cannot access private resources'. This rule can be expressed as follows:

$$p_2 : \mathit{private}(x), \mathit{blacklist}(y) \Rightarrow O_y \neg \mathit{access}(x)$$

Now to be effective this rule must be at least as strong as the rule to grant access (i.e, rule p_1). Thus we further need that $p_2 \prec p_1$.

3) Policies for Violations: The conditions that 'each member has to upload a picture to the profile page, and this picture is available to everybody in the network' and 'members who do not comply with the above conditions cannot access other members' private resources', can be expressed by the rule:

$$r_3: \Rightarrow O_x publish\_public \otimes O_x \neg access(y)$$

This rule establishes that a member has the obligation to publish at least one picture (visible to everybody), otherwise, the member cannot access any private resource. Accordingly, in case the first condition, i.e., $O_x publish\_public$, is violated, meaning that we have $\neg publish\_public$ (meaning that there are no public pictures). Then we can trigger the compensation (the sanction preventing the member to access other's member private resources). Notice that then the rule is in conflict with r_1 and p_1. So to make it effective we have to specify that $r_3 \succ r_1$ and $r_3 \succ p_1$.

C. Discussion

The reasoning mechanism presented above is to determine the obligations, permissions and prohibitions in force for a particular situation. The mechanism is based on the proof conditions of defeasible logic, which offers a constructive proof theory [**?**], [**?**]. The important aspect is that the constructive proof theory allows us to look at the derivation of a conclusion and to see the rules and facts used to derive a conclusion, and thus we can provide a full justification of why we obtained a specific outcome [**?**]. This feature caters for the accountability of our approach to policies.

Closely related is the efficiency and scalability. We envision that a policy server for a social network should be able to handle a large amount of requests. Thus the efficiency of the reasoning mechanism is of paramount importance. The outcome of an FCL policy can be computed in linear time [**?**]. Efficient implementations exist, and the most recent implementations fully support Semantic Web standards (RDF, RuleML) [**?**], and extensions to deal with normative conditions have been proposed [**?**].[3]

Defeasible logic and FCL have been proposed and applied for the representation of different aspects of normative reasoning (modelling contracts [**?**], modelling complex normative notions such as normative power and delegation [**?**], modelling norm changes [**?**], and business process compliance [**?**], [**?**]). Thus the principles on which FCL is based on seem to offer a faithful and conceptual representation of normative concepts such as those required to model policies in social networks. Therefore FCL offers a transparent framework for the domain under analysis.

In addition to the reasoning mechanism to derive the normative requirements in force, FCL has other reasoning mechanisms. For example, it is equipped with mechanisms to generate normal forms for a set of rules [**?**]. The normal form makes explicit all rules that can be obtained by combining given rules, and then removes redundant rules, i.e., rules whose meaning is included in the meaning of other rules. The generation of a normal form has several advantages for the design of policies. Since all conditions are made explicit, then it is possible to identify conflicts between policy condition, and at the same time it is possible to have a complete picture for the terms and conditions of a policy [**?**]. Accordingly, normal forms accounts for both the transparency and coverage of a policy.

VIII. Mappings between ORDL and FCL

We briefly discuss the mapping between ODRL and FCL, in particular we examine the correspondence between classes in ODRL and elements of FCL. In FCL we have three categories of objects: propositions, deontic operators and rules. Atomic propositions are built from predicates plus terms (where a term is either a variable or a constant), where variables and constants denote elements of the domain. For ODRL and social networks the domain of individuals consists of the resources and members of the social networks. Thus we can establish a mapping between the ODRL classes **Asset** and **Party** and individuals in FCL. Predicates describe properties of element of the domain and relationship between them, thus an FCL theory modelling the policy of a social network has to provide predicates corresponding to the attributes of the classes and to the relationships between classes. Thus for example the *Rightholder* relationship between the classes **Asset** and **Party** is modelled by the predicate $owner(x,y)$. In addition we create a predicate for each instance of the class **Action**, and we create appropriate predicates for instances of the class **Constraint**.

The classes **Permission**, **Duty**, and **Prohibition** are mapped to the deontic operators P and O (and $O\neg$ for instances of **Prohibition**). As we have seen in Section **??** the deontic operators bound literals (the literals should be a literal corresponding to one action, though in general in FCL a deontic operator can bind in general any type of proposition), and can be indexed by subject and beneficiary. These correspond to assignee and assignor in ODRL.

Finally a rule is a relationship between a set of instances of the class **Constraint** and an instance of any of the classes **Permission**, **Obligation** and **Prohibition**.

In Section **??** we have seen that FCL can handle violations, using the $\otimes$ operator, and the superiority relation ($\prec$), but these do not have counterparts in ODRL. This, however, is not a real limitation. As it was discussed in [**?**] the violation operator is useful for the design of the policy (i.e., when using the normal form to investigate the characteristics of a policy), but it is not required for the computation of the requirements. Indeed a rule $A \Rightarrow OB \otimes OC$ is equivalent to the following two rules $A \Rightarrow OB$ and $A, \neg B \Rightarrow OC$.

For the superiority relation, [**?**] shows how to take a set of rule with superiority theory in Defeasible Logic and transform it into an equivalent theory without superiority relation.

[3]A Java based open source of Defeasible logic and FCL is available at `http://spin.nicta.org.au/demo/SPINdle/`.

While the two constructors above are not required to specify the rule comprising a policy, we suggest that ODRL is extended to include elements corresponding to such notions to improve how policies are defined, relieving policy authors (where for social networks, for a private profile the designer can be the user itself) for the burned of specifying how to implement her policy conditions, and thus resulting in a more conceptual language for policies.

IX. Future Work

The work to date has focussed on developing an extensible Policy Language with the ODRL Version 2.0 Model and the Policy Expression challenge. There is still work to be performed on validating ODRL's expressibility and we plan to use new Social Network-based use cases from the FP7 PrimeLife Project [**?**]. We will also work on expressing the ODRL Model in RDF/XML to support Semantic Web applications of the ODRL policy language.

Next on the list is the Policy Transparency challenge and we plan to investigate mechanisms to best inform Social Network users on the terms of policies that apply to the content they are using. The Policy Conflict challenge will be the most formidable as it will push ontology matching to the extreme, but is where the Semantic Web technologies will provide the most benefit. For example, in the Alice use case, if one of the potential assignees has a privacy policy on their blog account that suppresses key information (e.g., the number of postings) then how can we deal with the inconsistency with the original policy?

Finally, the Policy Accountability challenge will bring the most acceptance of policies in Social Networks as we move away from the traditional enforcement regimes and become consistent with society norms. Such as allowing for freedoms on Social Networks but making sure the user is aware of the consequences of their actions (or inactions). We don't wish to preclude the possibility of policy violations and at the same time provide a trusted experience for all Social Network users. A key challenge for all the community.

X. Conclusion

Social Networks provide a unique circumstance now for the Policy and Semantic Web communities to apply previous research outcomes and implementation experiences at a scale not yet seen. The results will be significant but must address the needs of users first otherwise the uptake of these technologies will be a lost opportunity. We have shown some promising beginnings with a widely-deployed rights language that has evolved to become a general policy language. We hope that the future extensions and iterations of the language, and supporting systems, will rise to meet the challenges of the burgeoning policy-oriented semantic web.

Acknowledgements

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program and the Queensland Government.

References

[1] A. Mikroyannidis, "Toward a social Semantic Web," *IEEE Compute*, pp. 113–115, November 2007.

[2] J. Breslin and S. Decker, "The future of social networks on the internet," *IEEE Internet Computing*, pp. 86–90, Nov/Dec 2007.

[3] R. Iannella, "Industry challenges for social and professional networks," in *W3C Workshop on the Future of Social Networking*, Barcellona, 15–16 January 2009, http://www.w3.org/2008/09/msnws/papers/nicta-position-paper.pdf.

[4] G. Tonti, J. Bradshaw, R. Jeffers, R. Montanari, N. Suri, and A. Uszok, "Semantic Web languages for policy representation and reasoning: A comparison of Kaos, Rei, and Ponder," in *2nd International Semantic Web Conference (ISWC2003)*, ser. LNCS, no. 2870. Springer, 2003, pp. 419–437.

[5] S. Guth and R. Iannella, *ODRL V2.0 — Core Model. ODRL Initiative. Draft Specification: 16 January 2009*, http://odrl.net/2.0/DS-ODRL-Model.html.

[6] A. Passant, P. Kärger, M. Hausenblas, D. Olmedilla, A. Polleres, and S. Decker, "Enabling trust and privacy on the social web," in *W3C Workshop on the Future of Social Networking*, Barcellona, 15–16 January 2009, http://www.w3.org/2008/09/msnws/papers/trustprivacy.html.

[7] G. Governatori, "Representing business contracts in RuleML," *International Journal of Cooperative Information Systems*, vol. 14, no. 2-3, pp. 181–216, 2005.

[8] G. Governatori, Z. Milosevic, and S. Sadiq, "Compliance checking between business processes and business contracts," in *10th International Enterprise Distributed Object Computing Conference (EDOC 2006)*. IEEE Computing Society, 2006, pp. 221–232.

[9] G. Antoniou, D. Billington, G. Governatori, and M. Maher, "Representation results for defeasible logic," *ACM Transactions on Computational Logic*, vol. 2, no. 2, pp. 255–287, 2001.

[10] ——, "Embedding defeasible logic into logic programming," *Theory and Practice of Logic Programming*, vol. 6, no. 6, pp. 703–735, 2006.

[11] G. Governatori and A. Rotolo, "Logic of violations: A Gentzen system for reasoning with contrary-to-duty obligations," *Australasian Journal of Logic*, vol. 4, pp. 193–215, 2006.

[12] J. Carmo and A. J. Jones, "Deontic logic and contrary to duties," in *Handbook of Philosophical Logic, 2nd Edition*, D. Gabbay and F. Guenther, Eds. Dordrecht: Kluwer, 2002, vol. 8, pp. 265–343.

[13] D. Nute, "Defeasible logic," in *Handbook of Logic in Artificial Intelligence and Logic Programming*, 1994, vol. 3.

[14] G. Antoniou, A. Bikakis, M. Dimaresis, M. Genetzakis, G. Georgalis, G. Governatori, E. Karouzaki, N. Kazepis, D. Kosmadakis, M. Kritsotakis, G. Lilis, A. Papadogiannakis, P. Pediaditis, C. Terzakis, R. Theodosaki, and D. Zeginis, "Proof explanation for a nonmonotonic semantic web rules language," *Data & Knowledge Engineering*, vol. 64, no. 3, pp. 662–687, 2008.

[15] M. Maher, "Propositional defeasible logic has linear complexity," *Theory and Practice of Logic Programming*, vol. 1, pp. 691–711, 2001.

[16] N. Bassiliades, G. Antoniou, and I. Vlahavas, "A defeasible logic reasoner for the Semantic Web," *International Journal on Semantic Web and Information Systems*, vol. 2, pp. 1–41, 2006.

[17] G. Governatori and A. Rotolo, "A computational framework for institutional agency," *Artificial Intelligence and Law*, vol. 16, no. 1, pp. 25–52, 2008.

[18] G. Governatori, A. Rotolo, R. Riveret, M. Palmirani, and G. Sartor, "Variants of temporal defeasible logic for modelling norm modifications," in *Proceedings of 11th International Conference on Artificial Intelligence and Law*, 2007, pp. 155–159.

[19] S. Sadiq, G. Governatori, and K. Naimiri, "Modelling of control objectives for business process compliance," in *BPM 2007*, ser. LNCS, no. 4714. Springer, 2007, pp. 149–164.

[20] G. Governatori and Z. Milosevic, "A formal analysis of a business contract language," *International Journal of Cooperative Information Systems*, vol. 15, no. 4, pp. 659–685, 2006.

[21] C. Bournez and G. Neven, *Draft Requirements for Next Generation Policies. PrimeLife Deliverable H5.1.1, 11 December 2008*, http://www.primelife.eu/images/stories/deliverables/h5.1.1-policy_requirements-public.pdf.