Open Social Systems

Nardine Osman[1], Carles Sierra[1], Ronald Chenu-Abente[2], Qiang Shen[3], and Fausto Giunchiglia[2(✉)]

[1] Artificial Intelligence Research Institute (IIIA-CSIC), Barcelona, Spain
{nardine,sierra}@iiia.csic.es
[2] Dipartimento di Ingegneria e Scienza dell'Informazione, Univerità di Trento, Trento, Italy
{chenu,fausto}@disi.unitn.it
[3] College of Computer Science and Technology, Jilin University, Changchun, China
shenqiang19@mails.jlu.edu.cn

Abstract. While normative systems have excelled at addressing issues such as coordination and cooperation, they have left a number of open challenges. The first is how to reconcile individual goals with community goals, without breaching the individual's privacy. The evolution of norms driven by individuals' behaviour or argumentation have helped take the individual into consideration. But what about individual norms that one is not willing to share with others? Then there are the ethical considerations that may arise from our interactions, such as, how do we deal with stereotypes, biases, or racism, or how to avoid the abuse of community resources. This paper is concerned with accounting for individual needs while respecting privacy and adhering to the community's ethical code. We propose a decentralised architecture for normative systems that, along with the community norms, introduces individual's requirements to help mediate the interaction between members.

Keywords: Normative systems · Privacy by design

1 Introduction

Normative systems have attracted a lot of attention in the multi agent systems community as one approach to maintain the autonomy of agents while ensuring community goals and aspirations are fulfilled. Norms essentially specify the rules of interaction: what one can (or cannot) do, when, under what conditions, etc. Normative systems copy how human societies function, and they can be compared to social norms that govern society's behaviour or organisational norms that mediate interactions in organisations [8].

While normative systems have excelled at addressing issues such as coordination and cooperation [1], they have left a number of open challenges. The first is how to reconcile individual goals with community goals, without breaching the individual's privacy. A number of approaches have been studied to take the individual into consideration, such as norm synthesis techniques that would help

N. Bassiliades et al. (Eds.): EUMAS 2020/AT 2020, LNAI 12520, pp. 132–142, 2020.
https://doi.org/10.1007/978-3-030-66412-1_9

norms evolve based on individuals' behaviour [6], or norm evolution that would allow the individuals to reason about norms through argumentation [7]. But what about individual norms that one is not willing to share with their fellow community member? For example, imagine a community norm that states that a donation cannot be below 5€ and an individual norm that states that a donation cannot exceed 50€. Another open challenge are the ethical considerations that may arise from our interactions, such as, how do we deal with stereotypes, biases, or racism, or how to avoid the abuse of community resources, to name a few.

In other words, the question this paper addresses is how can we make sure that an individual will have their needs taken into consideration while we ensure their privacy is respected and the community's ethical code is not violated. To address these issues, this paper proposes a decentralised architecture for normative systems that, along with the community norms, introduces individual's requirements to help mediate the interaction between members. Section 2 presents our proposal in brief, Sect. 3 introduces the notation used in this paper, Sect. 4 introduces the decentralised architecture addressing the challenges discussed above, while Sect. 5 provides a motivating example, before concluding with Sect. 6.

2 Proposal

To address the issues presented above, we first say that in addition to community norms, there are also individual norms that describe the individual's rules of interaction with others.

Norms, as illustrated earlier, specify what actions are acceptable for that specific individual, who can the individual interaction with, and under what circumstances. While normative systems have focused a lot on the action, 'what' can one do, we highlight in this paper the other crucial aspect of interactions: 'who' can one interact with. The 'who' aspect has been implicit until now, usually hidden under the 'what' action specification. In an increasingly hyperconnected world, we choose to make the 'who' more explicit in our proposal. To achieve this, we require users to have profiles describing them, such as describing their gender, their age, their relationships, etc. With such profiles, rules on who to interact with can then be specified. For example, one individual norm can then say 'only seek the support of female friends during my breakup period', while another can say 'never ask my ex-husband for help'. As such, and in addition to community norms and individual norms, the individual profile becomes another crucial element for mediating our interactions.

Both the individual's norms and profile may be divided into a private and shared part. In what follows, we present the norms and the profiles in more detail.

2.1 Norms

As per the above we distinguish between community norms and individual norms.

– **Community norms**. These norms are the community's agreed upon norms. Any action (represented by a message exchange) in the peer-to-peer network must be coherent with them. We consider an action acceptable by the community when it doesn't violate any of the community's norms.
We note community norms can be categorised into a number of groups (Fig. 1). For example, institutional norms can describe the rules of behaviour in the given community (following the concept of electronic institutions [2]). Governance norms can describe the rules of who has the right to change existing norms and how. Ethical norms can describe what is considered ethical and what actions are deemed unethical, and hence, unacceptable in the community. Incentivising norms can help provide incentives for community members to behave in a certain way, such as encouraging benevolent behaviour, say to help maintaining the community and fulfilling its objectives. One can even imagine re-using, adapting, or building on top of existing norms. For example, a new social network may re-use the institutional norms of an existing social network and adapt them to their community's particular needs.

Fig. 1. Community norms

– **Individual norms**. These norms represent particular aspects of the relationship of the human with her machine and with the community. For instance, a prohibition to pop-up a message during a siesta unless coming from a relative. Or one can filter messages coming from people that they do not deem trustworthy. As most individual norms are private, some 'unethical' behaviour may be codified at this level and remain unnoticed, such as a private norm requiring to never show messages coming from old men.
In general, individual norms may implement certain behaviour that may not be fully aligned with the community values and norms. In cases of conflict between community norms and individual private ones, community norms prevail concerning actions within the community. For example, if community norms prohibit discriminating against women, then an action like excluding females from a given activity will be prohibited. However, individual private norms prevail when concerning actions local to one's machine. For instance, while community norms may prohibit discriminating against women, one's private norms can enforce requests coming from women to be suppressed

(ignored).
We note that individual norms can further be divided into two parts: private norms and shared norms.

- **Private norms** are norms that are private and are never shared with other machines (e.g. 'never show messages coming from old men'). Their impact is restricted as other machines do not have access to these norms.
- **Shared norms** are norms that travel with messages so that other people's machines can take them into consideration (e.g. when specifying 'do not ask the help of people outside Barcelona', the receiving machine can check the location of its human, even if this data is private as this data never leaves the machine and is not shared with others).

2.2 Profiles

Generally speaking we assume we have two types of profiles that we can intuitively describe as follows.

- **Private profile.** This is the set of features that are private to (and hence, accessible only by) the human's own machine. For instance, if `gender("A",female)` is part of Alice's private profile this means that Alice's machine has permission to use Alice's gender in the reasoning.
- **Shared profile**. This is a set of features that can be shared with (or made accessible to) others, both the humans and their machines. There are several approaches, both centralised and decentralised, that one can choose from for making information public. However, in this proposal, we suggest sharing the public profile by communicating it to other machines on an as-needed basis.

Of course, humans decide what part of their profile is public and what part is kept private.

The notion of private profile is quite intuitive. We want to keep private what we do not want the others to know. This issue of privacy has always been around but it has become of paramount importance with the pervasive use of the Web and the Social Media. In the past we were protected for free by our space and time limitations: it would take some time to from place A to place B and this time would increase with distance. The phone lifted some time barriers, but the propagation of information would still be limited by the fact that we were able to choose who to interact with and, in any case, the communication would only happen in pairs. Television lifted other barriers, allowing for zero time one-to-many communication, but still information was very much verified and under control and in many cases regulated by law. The Social Media have lifted the last barrier: now everybody can talk with everybody and say whatever they prefer with basically no limitations (the first limitations being established by the most recent legislation, for instance, GDPR in Europe).

The social media have made it possible to replicate and hugely expand what has always been the case in the real world. Now anybody can share information with anybody, virtually the entire world population, in zero time and no space

constraints. This motivates the focus on privacy and hence the need for a private profile.

But this is only part of the story. First of all, *the notion of privacy is not an absolute notion*. There is information that I may be willing to share with my family but not with my friends and even less with my enemies. For example people are usually very happy to share information about the location of their children in a certain moment of time, for instance the fact that they go to a school with a certain address and that lectures will end at 1pm, with a person with a car that maybe has a child who goes to the same school. But they would never be willing to share this information with a person they do not fully trust. In social relations, the notion of *privacy is fully contextual* in the sense that it depends on the current situation and also in the objectives that one wants to achieve.

The contextuality, and therefore non-absoluteness, of privacy brings up the key observation which underlies the need for both a public and a private profile. To provide another example which integrates the one about the child who needs to be picked up from school, suppose I have a certain disease, e.g., diabetes. This is sensitive information, namely information with many more constraints for its circulation. In general, most people would not talk about their disease, but, for instance, a person with diabetes, if too low in her level of sugar in the blood, would be very happy to let others know about this. And not only of the need for sugar but also of the fact that the reason is diabetes, as this would increase the urgency of the intervention. In social relations there is always *a tension between privacy and transparency*. In almost any interaction with other people we trade-off some privacy (about us, about our family, friends, ..., anybody) as a key enabler for obtaining information, support, information from others.

The notion of public profile captures exactly this need of transparency, meaning by this the sharing information as key to enabling social interactions. Clearly, the public profile is *contextual*, where the person we interact with is a crucial component of the relevant context, and mostly *dynamic*. There is in fact very little information, if any, that we are willing to always share with others; maybe our name, but also in this case it is easy to think of exceptions. Furthermore the public profile, like the private profile, will change in time because of multiple reasons, e.g., change of one's job or of the place where one lives. The contextuality and dynamicity of the public profile will require its continuous update and revision. This consists of a process which will be enforced by the local peer, as driven by its user, and which will consist of performing a set of abstraction operations [4] on the private profile.

3 Notation

We first present, in this Section, the notation used in the remainder of this paper. We say let CN describe the set of community norms, PrR and ShR describe the sets of private and shared norms, respectively, and PrP and ShP describe the private and share profiles, respectively. We view a profile as a set of features. To

specify which agent does a set of norms or profile describe, we use the sub index of that agent. For example, PrR_A describe's A's private norms whereas ShP_B describes B's shared profile.

We say a profile is a set of features, and we specify features as propositions. For example, we say *g*ender("A",female) to state that A's gender is female and *l*oc("A",barcelona) to state that A's location is in Barcelona. As for norms, we specify these as "if then" statements that talk about actions (similar to the rule-based norms of [3]), and we use the deontic operators O and F to describe obligations and prohibitions, accordingly. For example, F(display("A",M)) states that it is forbidden to display the message M to A.

4 Architecture and Associated Operational Model

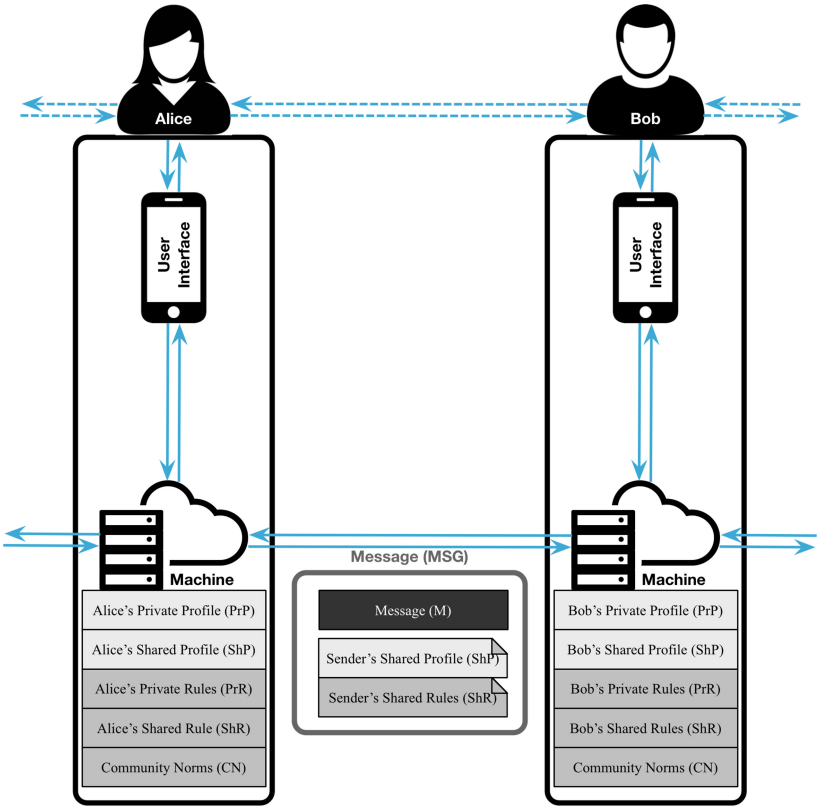

Fig. 2. Basic (distributed) architecture

In Fig. 2, the schema of the peer-to-peer architecture for our proposed normative system is presented. Each user has a machine, that may run all or some of its

computations on a remote server (depending on the complexity of the norms and their computational requirements). Each user interacts with its machine through a user interface.

As illustrated in Sect. 2, each user specifies their profile and individual norms. The profile is divided into private (PrP) and shared (ShP) parts, and the norms into private (PrR) and Shared (ShR) parts.

The norm engine at each machine will have both a reactive and proactive behaviour.

- **Reactive Behaviour.** This allows the norm engine to react to messages received (usually representing the actions being performed), and there are two types of messages that a machine can receive:
  - *A message from the user interface.* When a user performs an action, it is translated into a message that is sent to the machine through the user interface. The message includes the shared norms and a copy of the sender's shared profile. Upon the receipt of such a message, the norm engine needs to first verify that the message does not violate any of the norms, this includes the community norms and the sender's individual norms (both private and shared). A conflict resolution mechanism should address any conflicting norms that may arise. If the action violates any of those norms, an error message is sent back to the user. However, if the action obeys the norms, then the norm engine needs to decide what to do next, usually translated into sending messages to other peers. This decision follows from the community and individual norms (both private and shared), and takes the user's profile (both public and shared) into account as needed.
  - *A message from another machine.* As in the previous case, the norm engine needs to first verify that the message does not violate any of the community norms. This re-checking upon receipt ensures that the sender's norm engine has not been manipulated to cheat. If the message violates any of the community norms, then it may either be discarded, or if the community norms require sanctioning, then the appropriate sanctions should be executed.
    However, if the action obeys the community norms, then the norm engine needs to decide what to do next, which is usually translated into sending messages to other peers and/or sending messages to the user interface. This decision takes into consideration the community norms, the norms attached to the message, and the individual private and shared norms. This ensures that the machine abides with its human's private norms without leaking any of their private norms and profile.
- **Proactive Behaviour.** This allows the norm engine to proactively perform actions as required by the norms. For example, incentivising norms might remind a user to complete their profile, if this has been neglected for some time, or remind the user of how much their contribution to their community is valued, if they haven't been active lately. To be proactive, a machine will require access to the community norms and individual private norms, as well as its human's private and public profile.

5 Motivating Example

In this example we will specify the interaction between three people with the uHelp use case in mind. uHelp [5] is an app that allows one to find help with everyday tasks, such as picking up one's child from school, or finding a friend to play squash with. uHelp works by crawling one's social network looking for trusted volunteers. In this example, imagine having four people involved: Alice (A), Bob (B), Carol (C), and Dave (D). Say Bob, Carol and Dave are on Alice's contact list, and Bob is on Carol's contact list. The community norms (CN), or the uHelp norms, specify how a help request is propagated in the social network. They state that every time a machine receives a help request, it needs to decide whether it displays it to its user (Lines 32–35, Fig. 3), and whether it needs to forward it and to whom (Lines 24–31, Fig. 3). We note that the person making the request will decide the maximum number of hops accepted when looking for volunteers (`Hops`) and the minimum trustworthiness required (`Trust`). As these are specified for a given task, they are sent along with the help request message (Line 38, Fig. 3).

As for individual norms, imagine Carol has a private norm that says ignore help requests from females (i.e. do not display such requests, as show in Lines 13–17, Fig. 3). Alice, on the other hand, has a private norm and a shared one. The private one specifies that only those who are close by (in Barcelona) get to see her help requests (Lines 10–12, Fig. 3). The shared one specifies that none of her requests may be displayed to Bob (Lines 19–22, Fig. 3). As Bob is her ex-husband, she prefers that Bob does not see her requests, though she is happy for his machine to still receive her requests as she is interested in using his social network. Hence, she only prohibits the display of the message to Bob.

Now concerning people's profiles, some information such as gender, location, or trust in others may be kept private (Lines 1–4, Fig. 3), or made public (Lines 6–8, Fig. 3). For example, Alice's private profile specifies her trust in her contacts: in this case, 'low' for Bob and 'high' for Carol and Dave. Similarly, Carol's private profile specifies her trust in her contact Bob as 'high' and her current location as being in London. Bob, Dave and Alice are happy to make share their gender and location with others through their shared profiles.

Now given these profiles and norms, imagine that Alice is running late at work and she needs someone to pick up her child from school (message M). She accepts friends of friends (`Hops=2`, for connection level 2), but is looking for trustworthy volunteers only (`Trust="high"`), as illustrated in Line 38, Fig. 3. For these norms to be enforced by other machines, Alice shares these norms along with her other shared norms and profiles by attaching them to the help request (M), resulting in the message, MSG.

As soon as the help request is sent by Alice, the norm interpreter at her machine will check whether the message applies with the community norms (CN), in which case it does. The interpreter then needs to decide what are the actions that this message entails, taking into consideration Alice's profile (private and shared), her norms (private and shared), and the community norms. According to the community norm on Lines 24–31, the interpreter decides to

forward the help request to Carol and Dave, as they satisfy the requested hops and trustworthiness constraints (the trustworthiness of Bob, on the other hand, is low).

Upon receiving the message, Dave's machine now needs to check whether the message applies with the community norms, which it does. It then needs to decide what are the resulting actions of receiving this message, taking into consideration Dave's profile and norms (both private and shared), the norms attached to the message (that is, Alice's shared norms), and the community norms. According to the community norm on Lines 32–35, the request is then displayed to Dave, despite the fact that Alice forbids it in its private norm. This is because Alice's private norm is private and cannot be taken into consideration by other people's machines.

Upon receiving this message, again, Carol's machine needs to check whether the message applies with the community norms, which it does. After that, it needs to decide what are the resulting actions of receiving this message, taking

```
1   PrP_A = { trust("A","B",low) ;
2             trust("A","C",high) ;
3             trust("A","D",high) }
4   PrP_C = { trust("C","B",high) ; loc("C",london) }
5
6   ShP_A = { gender("A",female) ; loc("A",barcelona)}
7   ShP_B = { gender("B",male) ; loc("B",barcelona) }
8   ShP_D = { gender("D",male) ; loc("D", rome) }
9
10  PrR_A = { IF ¬loc(X,barcelona) ∧ requester(M)="A" THEN
11              F(display(X,M))
12            END IF }
13  PrR_C = { IF rcv_help_rqst(Sndr,"C",Trust,Hops,MSG) ∧
14                gender(Sndr)=female ∧
15                MSG=M+ShR_M+ShR_Sndr+ShP_Sndr THEN
16                    F(display("C",M))
17            END IF}
18
19  ShR_A = { IF rcv_help_rqst(Sndr,"B",Trust,Hops,MSG) ∧
20                MSG=M+ShR_M+ShR_Sndr+ShP_Sndr THEN
21                    F(display("B",M))
22            END IF }
23
24  CN = {IF rcv_help_rqst(Sndr,Rcvr,Trust,Hops,MSG) THEN
25           FOR ALL X∈Rcvr.Contacts
26             IF trust(Rcvr,X,Y) ∧
27               Y≥Trust ∧ Hops>0 THEN
28                 O(snd_help_rqst(Rcvr,X,Trust,Hops-1,MSG))
29             END IF
30           END FOR
31        END IF ;
32        IF rcv_help_rqst(Sndr,Rcvr,Trust,Hops,MSG) ∧
33          type(Sndr)=machine ∧ MSG=M+ShR_M+ShR_Sndr+ShP_Sndr THEN
34            O(display(Rcvr,M))
35        END IF }
36
37  M = "Can you pick up my son from school at 5pm?"
38  ShR_M = { O(snd_help_rqst("A",machine("A"),"high",2,MSG_M)) }
```

Fig. 3. uHelp example: profiles and norms

into consideration Carol's profile (private and shared), her norms (private and shared), the norms attached to the message (that is, Alice's shared norms), and the community norms. In this case, and according to Carol's private norm on Lines 13–17, the help request is not displayed on Carol's mobile as it comes from a female. However, the help request is forwarded to Bob, according to the community norm at Lines 24–31. Note that while Alice's trust in Bob was low, her trust in Carol is high, and Carol's trust in Bob is also high, allowing the message to be forwarded to Bob through Carol.

Upon receiving the message, Bob's machine again checks its adherence to community norms. Then, as above, it needs to decide what are the resulting actions of receiving this message, taking into consideration Bob's profile (private and shared), his norms (private and shared), the norms attached to the message (that is, Alice's shared norms), and the community norms. In this case, and according to Alice's shared norm on Lines 19–22, the help request is not displayed on Bob's mobile as Alice forbids it.

This example illustrates how our proposed system ensures the interaction between people adheres to both community norms and individual ones without jeopardizing people's privacy. It also illustrates the impact of private and shared information. For instance, private norms are better suited to control local behaviour, whereas shared norms are better suited for controlling the behaviour of other machines.

6 Conclusion

This paper has proposed a decentralised architecture for normative systems that introduces individual norms, while ensuring the privacy of people. One aspect that has been overlooked in this paper and left for future work is the conflict resolution mechanism. Having people specify their own norms will probably result in conflicting rules, and a mechanism will be needed to address such conflicts.

Our current next steps will be to implement the proposed system by extending the existing uHelp platform to introduce the different types of norms (adding private and shared ones) and different types of profiles (splitting them into private and shared). Furthermore, we plan to integrate uHelp with an extended version of iLog [9] that automatically learns people's profiles from their online activity.

As illustrated in our discussion of community norms, these norms can be used to specify the rules of interaction in a community, but also to introduce more specialised rules, such as rules specifying what is considered ethical and unethical, or rules specifying how to motivate people to act in a certain way. Future work will be experimenting with these specialised different, focusing on ethics and incentives.

Acknowledgements. This research has received funding from the European Union's Horizon 2020 FET Proactive project "WeNet – The Internet of us" (grant agreement

No 823783), as well as from the Spanish Ministry of Economy, Industry and Competitiveness' Retos 2017 project "CIMBVAL" (project No TIN2017-89758-R), and the RecerCaixa 2017 project "AppPhil".

References

1. Andrighetto, G., Governatori, G., Noriega, P., van der Torre, L.W.N. (eds.): Normative Multi-Agent Systems, Vol. 4. Dagstuhl Publishing (2013)
2. d'Inverno, M., Luck, M., Noriega, P., Rodriguez-Aguilar, J., Sierra, C.: Communicating open systems. Artif. Intell. **186**, 38–94 (2012)
3. García-Camino, A., Rodríguez-Aguilar, J.A., Sierra, C., Vasconcelos, W.W.: Constraint rule-based programming of norms for electronic institutions. Autonom. Agents Multi-Agent Syst. **18**(1), 186–217 (2009)
4. Giunchiglia, F., Walsh, T.: A theory of abstraction. Artif. Intell. **57**(2–3), 323–389 (1992)
5. Koster, A., et al.: U-help: supporting helpful communities with information technology. In: Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems, pp. 1109–1110. AAMAS 2013, International Foundation for Autonomous Agents and Multi-agent Systems, Richland, SC (2013)
6. Morales, J., López-Sánchez, M., Esteva, M.: Using experience to generate new regulations. In: Proceedings of IJCAI 2011, pp. 307–312 (2011)
7. Oren, N., Luck, M., Miles, S., Norman, T.: An Argumentation Inspired Heuristic for Resolving Normative Conflict. Unknown Publisher (2008)
8. Shoham, Y., Tennenholtz, M.: On social laws for artificial agent societies: off-line design. Artif. Intell. **73**, 231–252 (1995). https://doi.org/10.1016/0004-3702(94)00007-N
9. Zeni, M., ad Zaihrayeu, I., Giunchiglia, F.: Multi-device activity logging. In: ACM International Joint Conference on Pervasive and Ubiquituous Computing, pp. 299–302. ACM (2014)