COIN ++ @ ECAI

For ECAI 2016, COIN merged with NorMAS to become COIN++.

COIN@ECAI 2016

The emergence of open socio-technical systems raises a range of challenges and opportunities for research and technological development in the area of autonomous agents and multi-agent systems. In particular, human expectations about software behaviour - justified or not - significantly affect the evolution of human attitudes towards, acceptability of and creation of trust in such systems. Consequently, mechanisms that transmit representations of human values and how software can make decisions that respect them, are potentially significant for the effective design and construction of mixed human/software open systems.

Coordination, organizations, institutions and norms are four key governance elements for such systems, and the COIN workshops constitute a space for debate and exploration of these four elements in the design and use of open systems.

We seek to attract high-quality papers and an active audience to debate mathematical, logical, computational, methodological, implementational, philosophical and pragmatic issues related to the four aspects of COIN.

Of particular interest for the workshop are those papers that articulate a challenging or innovative view.

COIN is ranked B on the CORE Conference Ranking list: http://portal.core.edu.au/conf-ranks/2160/

Workshop chairs

Julian Padget (University of Bath, United Kingdom), j.a.padget@bath.ac.uk
Ana Paula Rocha (University of Porto, Portugal), arocha@fe.up.pt

NorMAS

Norms are crucial for studying both human social behaviour and for developing distributed software applications. The term *norms* is deliberately ambiguous. We study and apply norms in the sense of being normal (conventions, practice), and in the sense of rules and regulations (obligations, permisions).

Normative systems are complex systems in which norms play a crucial role or which need normative concepts in order to describe or specify their behaviour. A normative multi-agent system combines models for normative systems (dealing for example with conventions, or obligations) with models for multi-agent systems (dealing with coordination between individual agents).

Norms have been proposed in multi-agent systems and computer science to deal with issues of coordination, security, electronic commerce, electronic institutions and agent organization. They have been fruitfully applied to develop simulation models for the social sciences. However, due to the lack of a unified theory, many researchers are presently developing their own ad hoc concepts and applications.

The aim of this workshop is to stimulate interdisciplinary research on normative concepts and their application.

Workshop Chairs

Joris Hulstijn (Delft University of Technology, the Netherlands), j.hulstijn@tudelft.nl

Gabriella Pigozzi (Université Paris Dauphine, France), gabriella.pigozzi@dauphine.fr

Harko Verhagen (Stockholm University, Sweden), verhagen@dsv.su.se

Serena Villata (I3S Laboratory, CNRS, France), villata@i3s.unice.fr

COIN ++ @ ECAI 2016 schedule

9.30 – 10.30 Keynote Ibo van der Poel

10.30 – 11.00 Break

11:00 - 13:00 Session 1

A manifesto for conscientious design of hybrid online social systems - Pablo Noriega, Harko Verhagen, Mark d'Inverno and Julian Padget

The Role of Values - Klara Pigmans, Huib Aldewereld, Neelke Doorn and Virginia Dignum

Habit formation and breaking: a support agent design as an extended mind - Pietro Pasotti, Birna M. van Riemsdijk and Catholijn M. Jonker *

"How Did They Know?" - Model-Checking for Analysis of Information Leakage in Social Networks Louise Dennis, Marija Slavkovik and Michael Fisher

13:00 – 14:00 Lunch

14.00 - 15.30 Session 2

Monitoring Opportunism in Multi-Agent Systems - Jieting Luo, John-Jules Meyer and Max Knobbout

Sanction recognition: A simulation model of extended normative reasoning - Martin Neumann and Ulf Lotzmann *

An Architecture for the Legal Systems of Compliance-Critical Agent Societies - Antonio Carlos Rocha Costa *

15.30 – 16.00 Break

16.00 - 17.30 Session 3

Towards a Distributed Data-Sharing Economy - Samuel Cauvin, Martin Kollingbaum, Derek Sleeman and Wamberto Vasconcelos

Modelling patient-centric Healthcare using Socially Intelligent Systems: the AVICENA experience - Ignasi Gomez-Sebastian, Javier Vazquez, Frank Dignum and Ulises Cortes

Using Petri Net Plans for Modeling UAV-UGV Cooperative Landing - Andrea Bertolaso, Masoume M. Raeissi, Alessandro Farinelli and Riccardo Muradore

17.30 - 18.30 Round table: future of COIN/NorMAS

* denotes NorMAS papers

A manifesto for conscientious design of hybrid online social systems

Pablo Noriega[1], Harko Verhagen[2], Mark d'Inverno[3], and Julian Padget[4]

[1] IIIA-CSIC, Barcelona, Spain pablo@iiia.csic.es
[2] Stockholm University. Stockholm, Sweden
verhagen@dsv.su.se
[3] Goldsmiths, University of London, London, UK
dinverno@gold.ac.uk
[4] Department of Computer Science, University of Bath, Bath, UK
j.a.padget@bath.ac.uk

Abstract. Online Social Systems such as community forums, social media, e-commerce and gaming are having an increasingly significant impact on our lives. They affect the way we accomplish all sorts of collective activities, the way we relate to others, and the way we construct are own self-image. These systems often have both human and artificial agency creating what we call online hybrid social systems. However, when systems are designed and constructed, the psychological and sociological impact of such systems on individuals and communities is not always worked out in advance. We see this as a significant challenge for which coordination, organisations, institutions and norms are core resources and we would like to make a call to arms researchers in these topics to subscribe a conscientious approach to that challenge.
In this paper we identify a class of design issues that need attention when designing hybrid online social systems and propose to address those problems using *conscientious design* which is underpinned by ethical and social values. We present an austere framework to articulate those notions and illustrate these ideas with an example. We also outline five lines of research that we see worth pursuing.

1 Introduction

We are witnessing major social changes caused by the massive adoption of online social systems that involve human users alongside artificial software entities. These hybrid online social systems promise to satisfy and augment our social needs and the rise of such systems and their use are nothing short of spectacular. Because of the speed of their uptake their has been limited research that looks at the relationship between system design and potential long-term psychological, sociological, cultural or political effects.

Examples of the undesirable consequences of such systems (with varying degrees of autonomous agency participation) include:

– the increasing importance of social media expressions and reactions in building and maintaining identity,

– the possibility of determining personal data from facial recognition applications such as *FindFace*,
– the possibility of determining personal information via automatic scrubbing of online dating services such as *OKCupid*,
– the everchanging algorithm for presenting messages on *Facebook*, outside of the control of the user

The social impact of these applications is magnified by the accessibility of mobile devices, ubiquitous computing and powerful software paradigms that enable innovations in AI to be readily integrated. Despite this, design takes place in an *ad-hoc* and opaque way so that the social consequences of online actions are unknown. The effect of online actions in the real social world is often not understood, we often do not know whether actions are private or public, we cannot be sure of the way in which the actions of others is presented to us, and nor do we know how information about our activity is being used.

As the AI community plays a key role as inventors and builders of the scientific and technological enablers of this phenomenon, we have a moral responsibility to address these issues that requires a sustained, long term commitment from our community. We believe that what is needed is a collective interdisciplinary endeavour across design, sociology, formal methods, interface design, psychology, cultural theory, ethics, and politics to develop a clearer understanding of how we approch and design online social systems. Together we can play an active role in the design of systems where users' understanding of actions, relationships and data is fair and clear. The challenge is great, but then so is the responsibility. Those of us working in the theory, design and implementation of agent-based systems now have a fantastic opportunity to apply our methods and tools in ways which could have impact far beyond that we might have imagined even a few years ago.

This paper then is *a call to arms* for such an initiative, specifically to the COIN community, in the spirit of the "Research Priorities for Robust and Beneficial Artificial Intelligence: an Open Letter". We articulate our proposal around the notion of *conscientious design* as a threefold commitment to a design that is *responsible*, *thorough* and *mindful*.[5].

Conscientious design starts by developing an awareness of the concerns manifest in the current landscape, and understanding how multi-agent techniques can be applied as an effective means to operationalise systems to ameliorate such concerns, and bring it to bear upon our everyday scientific and technological activity. For this we need to (further) develop theories and models of norms, roles, relationships, languages, architectures, governance and institutions for such systems, and do so in a way that naturally lends itself to interdisciplinary research. We need to be *empiricists* (in applying our techniques to modelling current systems), *theorists* (in building implementable models of hybrid social systems), and *designers* (in designing systems); open to working in a strong *interdisciplinary* way across arts, humanities and social sciences. We may also need to break away from our natural comfort zones describing idealised scenarios for

[5] `http://futureoflife.org/static/data/documents/research_priorities.pdf`

agents but we can do so when we recognise just how potentially significant the impact of our research can be.

In this paper we postulate the need to address this challenge, propose a focus of attention —Hybrid Online Social Systems (HOSS)— and give a rough outline of what we see as the main research questions. The paper is structured as follows: In Sec. 2 we point to some background references so as to motivate our election of problematic aspects of HOSS and our proposal of conscientious design, addressed in Sec. 3. In Sec. 4 we propose the core ideas —based on the WIT framework [15]— to make conscientious design operational and in Sec. 5 we illustrate these ideas with an example. All these elements are then put together as a research programme towards conscientious design and implementation of HOSS.

2 Background

2.1 The problem

The range of behaviours that we can carry out online make available all kinds of activity that was not possible even a few years ago. It can affect how we see ourselves, how we choose to communicate, how we value notions of privacy and intimacy, and how we see our value in the world. We are building new metaphors of ourselves while we are in contact with everyone and everybody [9]. The issue that is overlooked by many users is that almost anything that can happen in the real social world —i.e. the one which existed before online systems— can potentially happen in any online one, and worse. We are facing a "Collingridge dilemma": We do not yet know how to take advantage of the opportunities of this technology and avoid its unwanted consequences but we are justifiably concerned that by the time we understand its side-effects it may be too late to control them [6].

2.2 An approach to the solution

We concern ourselves with those systems where there is artificial agency; either because there are software socio-cognitive agents that have some autonomy or because the system infrastructure incorporates agency (such as by actively producing outcomes that are not the ones users expect, or because third parties may interact with that system without the system or its users being aware or intending it to happen). For these "hybrid online social systems", or HOSS, we identify the generic type of features we find problematic and propose a "conscientious" design approach in response.

Our proposal is in tune with the *Onlife Manifesto* [9] and thus aims to respond to the sensitivities and challenges captured in that document. For instance, a *new understanding* of values, new uses of norms and the new guises that their enforcement should take; attention to how values like trust, fairness, solidarity are understood; give users control over the way their own values may become incorporated in the tools they create or adopt. Our proposal can be framed as a part of the "value alignment problem".[6]

[6] Stuart Russell:"... *The right response [to AI's threat] seems to be to change the goals of the field itself; instead of pure intelligence,* we need to build intelligence that is provably aligned with human values...". `https://www.fhi.ox.ac.uk/edge-article/`

Our proposal is akin to the Value-sensitive design (VSD) research framework [10] and similar approaches like *Values in Design* [13] and *disclosive computer ethics* [3]. The main concern in VSD is how values are immersed (mostly unconsciously) in technological artifacts, and postulate that what is usually missing during the design and development phases is a critical reflection upon this unconscious inscription of values. We advocate a *conscientious* approach to put in practice that critical reflection.

VSD offers three "investigation" schemata for inscribing values into the design of systems (i) *conceptual-philosophical* whose aim is to identify relevant values, and relevant direct and indirect stakeholders (not only users), (ii) *empirical* the use of qualitative and quantitative research methods from the humanities and social sciences, to study how people understand and apply values, and (iii) *technical* to determine the role that values play in technologies and how to implement those values identified in the two previous schemata into the systems that are being designed.

We propose a narrower but complementary strategy. We propose to focus attention in those values that are associated with three broad areas of concern that we believe are encompassed by conscientiousness: *thoroughness* (the sound implementation of what the system is intended to do), *mindfulness* (those aspects that affect the individual users, and stakeholders) and *responsibility* (the values that affect others). We postulate an approach to software engineering that is directed towards a particular class of systems (HOSS). It is an approach close to VSD because it rests on a particular categorisation of values but we go further because we understand that those values are instrumented by means of institutional (normative) prescriptions that have an empirical and conceptual grounding, and then implemented through technological artifacts that have a formal grounding. Consequently, while from a teleological point of view we see our approach closer to the ideas of value-sensitive-design, from a technological and methodological point of view, the domain and the proposal are clearly within the COIN agenda.

2.3 The Role of COIN

We believe there is a critical need for a science and discipline of conscientious design for online hybrid social systems which contain human and computational entities. Some of the questions that present themselves to our community are given below.

- How can the agent/AI community collectively recognise this opportunity and spring into action to take part in the development of a science of hybrid online social systems (HOSS) that can lead to their principled design?
- How can we build models, tools, methods and abstractions that come from our own specialities across agent design, interaction protocols, organisations, norms, institutions and governance to underpin the principled design of software incorporating human and artificial agents?
- How can we encourage and support a greater degree of responsibility in the design of online environments in exactly the same way as an urban planner would feel when designing a new locale?

This is not an easy task as the domain is such a diverse and complex oneThis is necessarily an early foray into setting up the challenges of charting this space and defining

some of the challenges we face in order to do so and doing so in way in which we can build bridges to other communities. Naturally, we want any undertaking to be wide ranging, to be inclusive so that people from all fields of the agent and AI communities can take part, and where groups from other parties can join with a clear sense of what we mean by a science of online social systems. Studies from other disciplines often lead to important critiques of technological development, what *our community can uniquely provide is a scientific framework* for system design that can both critique current systems but also enable a collective design of future conscientious systems. We will all lose out if there cannot be a collective and interdisciplinary approach to understanding how to design such systems. We need a common technological and scientific framework and language to argue for how we should design the next generation of such systems.

3 Choice of problems and approach: conscientious design of HOSS

The first challenge we propose to address is to develop a precise characterisation of HOSS. As suggested in [5], this can be approached in two directions. First a bottom-up task that consists of studying existing HOSS to identify their essential features and typologies. For each typology we suspect there will be particular ways in which desired properties may be achieved. The task would be to elucidate how values like transparency, accountability, neutrality, and properties like hidden agency and such are achieved in the actual systems and look for those design and implementation resources that tell the degree to which those properties exist. Secondly, top-down research would aim to approximate agent-based abstract definitions of ideal classes of HOSS and gradually make them precise in order to *analytically* characterise the features and properties of the HOSS we design and build.

Far the moment we will speak of HOSS in not-formal terms from the top-down perspective. Loosely speaking, HOSS are IT enabled systems that support collective activities which involve individuals —human or artificial— that reason about social aspects and which can act within a stable shared social space.[7]

This is a tentative "analytic" definition of HOSS (from [15]):

Notion 1 *A* Hybrid online social ssytem (HOSS) *is a multiagent system that satisfies the following assumptions:*

A.1 ***System*** *A socio-cognitive technical system is composed by two ("first class") entities: a* social space *and the* agents *who act within that space. The system exists in the real world and there is a boundary that determines what is inside the system and what is out.*

A.2 ***Agents*** *Agents are entities who are capable of acting within the social space. They exhibit the following characteristics:*

[7] Such systems have been labelled "socio-technical" [20], *socio-cognitive technical systems* [4], *intelligent socio-technical systems* [12] and we called them *socio-cognitive technical systems* in [15].

A.2.1 ***Socio-cognitive*** *Agents are presumed to base their actions on some internal decision model. The decision-making behaviour of agents, in principle, takes into account social aspects because the actions of agents may be affected by the social space or other agents and may affect other agents and the space itself [4].*

A.2.2 ***Opaque*** *The system, in principle, has no access to the decision-making models, or internal states of participating agents.*

A.2.3 ***Hybrid*** *Agents may be human or software entities (we shall call them all "agents" or "participants" where it is not necessary to distinguish).*

A.2.4 ***Heterogeneous*** *Agents may have different decision models, different motivations and respond to different principals.*

A.2.5 ***Autonomous*** *Agents are not necessarily competent or benevolent, hence they may fail to act as expected or demanded of them.*

A.3 ***Persistence*** *The social space may change either as effect of the actions of the participants, or as effect of events that are caused (or admitted) by the system.*

A.4 ***Perceivable*** *All interactions within the shared social space are mediated by technological artefacts — that is, as far as the system is concerned there are no direct interactions between agents outside the system and only those actions that are mediated by a technological artefact that is part of the system may have effects in the system — and although they might be described in terms of the five senses, they can collectively be considered percepts.*

A.5 ***Openness*** *Agents may enter and leave the social space and* a priori*, it is not known (by the system or other agents) which agents may be active at a given time, nor whether new agents will join at some point or not.*

A.6 ***Constrained*** *In order to coordinate actions, the space includes (and governs) regulations, obligations, norms or conventions that agents are in principle supposed to follow.*

3.1 Our focus of attention: Hidden agency

The main problems with HOSS are what for a lack of a better term we'll call "unawareness problems" such as *hidden agency*, *insufficient stakeholder empowerment*, and *lack of social empathy*.

Perhaps more than anything, we need to draw out the extent to which these systems have or may acquire *hidden agency*. We mean, those side-effects or functionalities of the system that are exploitable by its owner or others without the user being fully aware of them, even if they were unintended by the designer of the system. In the language of multi-agent systems from 25 years ago, there is an assumption that the agency of online systems is benevolent [11] but if the hidden agency was revealed to users it would often be entirely unwelcome and unwanted. And in the same language, we may see hidden agency as hidden limits to the autonomy of the user.

An example of hidden agency is the recent case of mining on *OKCupid* where a group of researchers not only mined the data of the online dating service but even put the data collection of 70,000 users online on the Open Science Framework for anyone to use. Although real names were not included, the data of personal and intimate character could easily be linked to find the real identity behind the user names. Even

more so, if it would be connected via the profile pictures (which the researchers left out of the database due to space reasons, not ethical considerations) to other social media when using software such as Facefind (http://www.findbyface.com/) and Findface (http://www.findface.ru) Although *OKCupid* managed to have the data removed on copyright violations, in what way the users had an opinion on or say in this is very unclear (a case of insufficient stakeholder empowerment).

A case of lack of social empathy is how the use of *Facebook* for memorial pages may have distressing effects [17]. Large turn-ups at funerals offer comfort and support to those who have lost a loved one. The same effect also applies to online shows of mourning such as the deluge of messages posted when a famous person dies. They show up in the trending topics bar on *Facebook*, spreading the news fast. Even for less famous persons, *Facebook* is playing a role in the mourning process. *Facebook* pages are kept alive, messages are sent to the deceased and memorial pages are put online. But not all is good. Just as a low turn-up at a funeral will cast doubt on the legitimacy of ones sorrow so is the failure of attention in *Facebook* creating doubts. Moreover, the turn-up at a funeral is a private observation limited in time and space whereas *Facebook* measures and shows it all. The number of visitors can be compared to the number of likes or other *emojis* and the number of comments, for all to see.

3.2 What we mean by conscientious design

We will go beyond value-sensitive design towards conscientious design and development. As we mentioned in Sec. 2, we propose to look into a particular set of values —involving technical, individual and social domains— that are linked to the description, specification, implementation and evolution of HOSS. Thus conscientious design and developent of HOSS responds to three properties:

1. *Thoroughness.* This is achieved when the system is technically correct, requirements have been properly identified and faithfully implemented. This entails the use of appropriate formalisms, accurate modelling and proper use of tools.
2. *Mindfulness.* This describes supra-functional features that provide the users with awareness of the characteristics of the system and the possibility of selecting a satisfactory tailoring to individual needs or preferences. Thus, features that should be accounted for should include ergonomics, governance, coherence of purpose and means, identification of side-effects, no hidden agency, and the avoidance of unnecessary affordances.
3. *Responsibility.* This is true both towards users and to society in general. It requires a proper empowerment of the principals to honour commitments and responsiveness to stakeholders legitimate interests. Hence, features like its scrutability, transparency and accountability alongside a proper support of privacy, a “right to forget”; proper handling of identity and ownership, attention to liabilities and proper risk allocation, and support of values like justice, fairness and trustworthiness.

It is here the agent metaphor for system design provides a clear opportunity for providing models that can be understood by academics, users and designers of HOSS. For the commercial-driven applications we might think of designing conscientiousness

sensors, small apps that show warning flags when the online application in use collides with the values of the user. But in the remainder of the paper we will look at applications developed in a conscientious way and illustrate the points we wish to make by revisiting applications developed by or close to us.

4 An abstract understanding of HOSS

In order to design HOSS using a conscientious approach we need to come up with a clear characterisation of these systems. Eventually, we should be able to articulate a set of features that discriminate the online social systems that we are interested in — the ones with "unawareness problems" we mentioned — from other online social systems. In our research programme we propose to take a twofold approach for this task: an empirical, bottom-up line that starts from paradigmatic examples and a top-down line that provides an abstract characterisation. We already took a first step along this second line with the WIT framework proposal that we summarise here.[8]

We start from the observation that HOSS are systems where one needs to *govern* the interaction of agents that are situated in a physical or artificial world by means of technological artifacts. The key notion is "governance" because in order to avoid hidden agency and other unawareness problems we need to control on one hand, the frontier between the system itself and the rest of the world and, on the other, the activity of complex individuals that are at the root of HOSS. In order to elucidate how such governance is achieved we proposed the following tripartite view of HOSS (Fig. 1):

View 1: An *institutional* system, $\mathcal{I}$, that prescribes the system behaviour.
View 2: The *technological artifacts*, $\mathcal{T}$, that implement a system that enables users to accomplish collective actions in the real world ($\mathcal{W}$), according to the rules set out in $\mathcal{I}$.
View 3: The system as it exists in the *world*, $\mathcal{W}$, as the agents (both human and software) see it and with the events and facts that are relevant to it.

In other words, $\mathcal{W}$ may be understood as the "organisation" that is supported by an "online system" $\mathcal{T}$ that implements the "institutional conventions" $\mathcal{I}$.

Notice that we are referring to one single system but it is useful to regard it from these three perspectives because each has its own concerns. Notice also, these three perspectives need to be *cohesive* or "coherent" in a very particular way: at any given time t, there is a *state of the system* s_t that is exactly the same for all agents that are in the system, and when an agent interacts with the system (in $\mathcal{W}$), that state of the system changes into a new state s'_t, which is again common to all agents, if and when the agent's action is processed by the system (in $\mathcal{T}$) according to the specifications of the system (in $\mathcal{I}$).

In order to make this cohesion operational, we define three binary relations between the views. As sketched in Fig. 1, the institutional world *corresponds* with the real world by some sort of a "counts-as" relationship [19] —and a mapping between entities in

[8] See [15] for a more leisurely discussion of the WIT proposal.

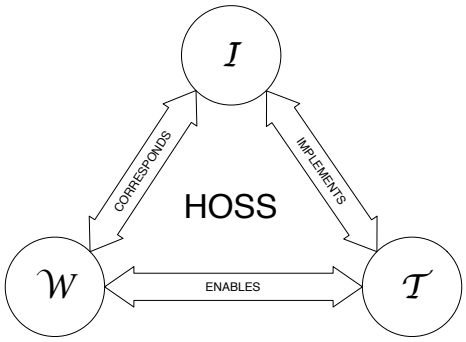


Fig. 1. The WIT trinity: The ideal system, $\mathcal{I}$; the technological artifacts that implement it, $\mathcal{T}$, and the actual world where the system is used, $\mathcal{W}$.

$\mathcal{W}$ and entities in $\mathcal{I}$— by which *relevant* (brute) facts and (brute) actions in $\mathcal{W}$ correspond to institutional facts and actions in $\mathcal{I}$ (and brute facts or actions have effects only when they satisfy the institutional conventions and the other way around). Secondly, $\mathcal{I}$ specifies the behaviour of the system and is *implemented* in $\mathcal{T}$. Finally, $\mathcal{T}$ *enables* the system in $\mathcal{W}$ by controlling all inputs that produce changes of the state and all outputs that reveal those changes.

It should be obvious that HOSS are not static objects. Usually, each HOSS has a lifecycle where the process of evolution is not all that simple [5].

4.1 A WIT understanding of conscientious design

Conscientious design adds meaning to the WIT description by throwing light upon certain requirements that the three binary relations should satisfy. Thus, in the first phase of the cycle, the main concern is to make the design value-aware from the very beginning, in line with the recommendations of value-sensitive-design. That is, analyse systematically the *thoroughness*, *mindfulness* and *responsibility* qualifications of the system, so those ethical, social and utilitarian values that are significant for the stakeholders are made explicit. This examination would then pursue a proper operationalisation of the intended values so that they may be properly translated into institutional conventions. Note that it is in this phase where mindfulness and responsibility analysis of requirements are more present, while thoroughness is the focus of the next stage.

As suggested in [15], the operationalisation of those values together with the usual software engineering elements (functionalities, protocols, data requirements, etc.) should be properly modelled (in $\mathcal{I}$) and then turned into a specification that is implemented in $\mathcal{T}$. The passage from the elicitation of requirements to the modelling of the system is facilitated by the availability of *metamodels* [1] that provide the *affordances* to represent correctly those requirements. Ideally, such representatio should satisfy three criteria: they should be *expressive*, they should be formally *sound* and it should become *executable*. The metamodel should also provide *affordances* to model the evolution of the system. Note that when relying on a "metamodel", its expressiveness will bias the way conscientiousness is reflected in the eventual specification.

The running system requires components for validation of the functionalities of the system, for monitoring performance and the devices to control transfer of information into and out of the system. These validation and monitoring devices should be tuned to the conscientious design decisions and therefore reveal how appropriate is the implementation of the system with respect to conscientious values and where risks or potential failures may appear.

5 How to achieve conscientious compliance

The abstract WIT and cosnscientious design ideas take rather concrete forms when building new HOSS.

5.1 An example of conscientious design, the *uHelp app*

Picture a community of *monoparental* families that decide to provide mutual support in everyday activities: baby-sitting, picking up children from school, go shopping, substitute at work during an emergency, lending each other things like strollers, a blender. One may conceive an *app* that facilitates such coordination. But —sensitive to conscientious design— one wants to make sure that coordination is in accordance with the values of the community. In this case, for example, *solidarity*: everyone helps each other for free; *reciprocity*: no free riding; *involvement*: old people may want to help; *safety*: no one without proper credentials should be able to pick up a child; *privacy* (no revelation of personal data, of behaviour of members of the network); *trust*: you demand more trustworthiness in some tasks than others and trust is a binary relation that changes with experience.

You program the *app* so that it reflects those values faithfully and effectively. Moreover, you want the community to be aware of the degree of compliance/usefulness of the network, and that the community may change the specification to improve it or adapt to new preferences or values. Also you want the *app* to be unobtrusive, reliable, practical (light-weight, easy to download, easy to support, easy to update), and not contain hidden agency.

Abstracting away from the actual specification, the main conscientious-compliance features that the *app* should have are:

1. *From a practical perspective:* (i) Useful for the relevant coordination tasks, (ii) Faithful and responsive to the community's goals, preferences and values, (iii) Have the community in control of evolution (iv) No hidden agency.
2. *From an institutional perspective:* (i) shared ontology, (ii) common interaction model and interaction conventions (the *smartphone app*), (iii) govern a core coordination process: values, norms, governance (iv) controlled evolution: participatory, reliable, effective, (v) no unwanted behaviour.
3. *From a technical perspective:* (i) proper monitoring (key performing indicators, historical logs), (ii) automated updating (iii) robust and resilient *app*. (iv) Safe against intrusions and "zero information transfer" (only the intended information is admitted into the system and only intended information is revealed).

This type of application and the conscientious-design perspective have been under development in the IIIA for some time [16], and there is a working prototype, *uHelp*, that implements these ideas in a *smartphone app* and has already undergone field tests with actual users [14].

Where in WIT is conscientiousness

This example also serves to illustrate how conscientious design considerations may be reflected in the WIT cycle:

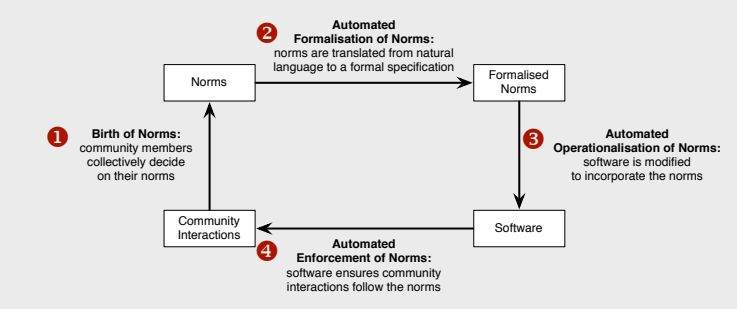


Fig. 2. Life-cycle of norms in the *uHelp app* from [16]

For specification: The *UHelp app* exists as a smartphone-based social network in $\mathcal{W}$. It involves two realms: The first one consists of the physical components of the system, which includes smartphones, addresses, schools, ID cards, blenders and strollers, as well as the organisation of parents that own the application and the group of technicians that support is everyday use and maintenance. The other is the activities that are coordinated with the *app* (picking children up, help with shopping) and the activities that are needed to use the *app* (running a server, uploading the *app* in *iTunes*). Thus in order to describe (in $\mathcal{I}$) how it should work, WIT would need an *expressive description language* that should include coordination conventions, values, norms, and so on. In other words, a description language that can handle *mindful* and *responsible* values. On the other hand, the specification should be such that users are comfortable with the conventions that govern the system and its evolution; and in this respect, the system needs to be *thorough*.

For formalisation: Description needs to be made precise: How are values associated with norms? Does the system support norm changes with some formal mechanism? Is simulation the appropriate tool for validation and monitoring? In our case, *UHelp* is intended to have a development *workbench* that uses electronic institutions coordination and governance affordances (an EI-like metamodel [8]) that is being extended to handle values. Furthermore, the *UHelp* workbench shall contain also an argumentation environment for arguing about normative changes (to empower stakeholders) and a simulation module to test and anticipate (responsibly) potential changes of the system.

For implementation: One would like to rely on technological artifacts that make a *thorough* implementation of the specification of the system. Those artifacts may include devices like model checking, agent-mediated argumentation, agent-based modelling and simulation. In particular, the *uHelp* workbench shall be coupled with a platform that deals with the implementation of the functionalities of the value-based social network and also with the implementation and maintenance of the *app* itself.

What does it mean to be *conscientious* in the *uHelp app*?

This is a sketch of an answer for a *uHelp*-like HOSS.

Thorough: For specification purposes, a metamodel that *affords* proper representation, sound formalisation, correct implementation of: (i) Coordination and governance (activities, communication, social structure, data models, procedural norms, enforcement, etc.) (ii) Values, (ontology, norms, inference) (iii) Monitoring (KPI, use logs) (iii) Evolution (automated or participatory updating, validation).

Mindful: Proper elicitation and operationalisation of *values*, preferences and goals, sensible selection of functionalities; lucid assessment of performance; explicit *stakeholders entitlements and responsibilities*; sensible attention to *usability and culturally sensitive* issues; due attention to *privacy.* What *agency* is afforded by the system?

Responsible: (i) Clear and explicit commitments about *information transfer* in the system, uses of performance data, and about *management* of the system. (ii) Clear requirements and commitments of system *updating*: what may users do; what type of guarantees and requirements are part of the evolution process. (iii) Proper description of coordination behaviour (requirements and outcomes for intended behaviour of automated activities and support functionalities). (iv) Explicit description about *ownership* of the system, about relationship with *third-party software* and about *commercial* and other commitments with *third parties*.

5.2 Three roads to application:

Rather than Quixotic fighting of *Facebook* windmills and trying to make existing HOSS conscientious-compliant we identify three lines of attack: (i) Conscientiousness by design, like the *uHelp* example; (ii) methods and devices to test the extent to which an existing HOSS is conscientious-compliant. This includes means to determine analytically whether a given HOSS has problems like hidden agency, insufficient user empowerment, inadequate social empathy; and (iii) *plug-ins* that may provide some conscientious-compliant features to existing HOSS.

6 Towards a new Research Programme

In order to support conscientious design, we propose a research programme (based on [15]) around the following five topics (see Fig. 3):

1. Empirical foundations: Conscientious design intends to build systems that support expected values and avoid unwanted features and outcomes. As we have been arguing in previous sections, we find that a systematic examination of actual socio-technical systems and of the values and unwanted outcomes involved need to be at the root of

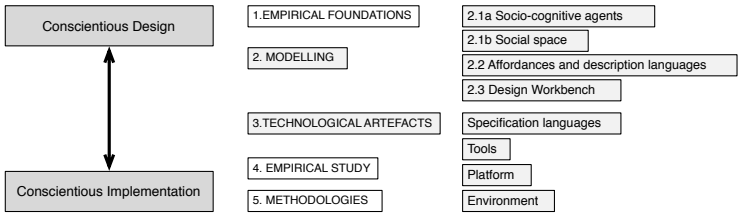


Fig. 3. The main challenges in the development of a framework for conscientious design of hybrid online social systems.

formal, technological and methodological developments in conscientious design. The outcomes should be, on one hand, a proper characterisation of HOSS and, on the other, a proper operationalisation of problematic manifestations in HOSS and the preventive and remedial features based on design conscientiousness.

2. Modelling: Conscientious design means: (i) that the creation of each HOSS be founded on a precise description of what the system is intended to be; (ii) that such description be faithfully implemented; and (iii) that the implementation actually works the way it is intended to work. In fact, it would be ideal if one could state with confidence the actual properties —scalability, accuracy, no unwanted side-effects, etc.— that the working HOSS has, because either we design the system with those properties in mind or because we are able to predicate them of an existing HOSS or an existing HOSS supplemented with *ad-hoc* plug-ins.

We propose to split the problem of conscientious modelling in three main parts: (2.1) Separate the design of a HOSS in two distinct concerns (the design of socio-cognitive agents and the design of a social space); (2.2) develop high-level description languages; and (2.3) develop a "design workbench" that provides concrete modelling components that translated the description of a HOSS into a specification.

2.1.(a) Socio-cognitive agents. First it is important to provide a conceptual analysis of the types of agents that may participate in a HOSS. The significant challenge is to create agent models that exhibit true socio-cognitive capabilities Next to it is the challenge of developing the technological means to implement them; hence the definition of agent architectures using a formal and precise set of agent specification languages with the corresponding deployment and testing tools.

2.1.(b) The social space. In addition one has to provide a sufficiently rich understanding of the social spaces which are constituted in HOSS. What are the relationships, what are the norms, how can it evolve, and a clarity about how this space is related to the external world. Any model would also need to consider how several HOSS may co-exist in a shared social space. Features that need to be included are openness, regulation, governance, local contexts of interaction, organisational and institutional structures.

2.2. Affordances and description languages. We need to identify the *affordances* that are needed, both, to achieve conscientious design in general, and also to support a *thorough* implementation of particular HOSS (as illustrated in Sec. 5). In other words, what are the concepts, analogies and expressions that a social scientist, an urban plan-

ner, a game designer or a sociologist may find more suitable to model agents and social space of a HOSS. In practice, a description language for modelling agents should afford the means for the agent to be aware of the state of the system, of its own state, and to hold expectations of what actions it and other participants can take at a given state. For modelling the social space, the language should be able to express those elements that *afford* participants the means to have a shared ontology, a common interaction model and communication standards coupled with some form of governance.

2.3. Design workbench. It would include the concrete versions of the affordances. That is, the "vocabulary" that the description languages will use in order to model an actual system. So, for instance, if the system will involve norms, then the workbench would have norms expressed with a particular structure together with concomitant paranormative components like normative inference, nor-enforcement mechanisms, etc. In the *uHelp* example, we need functional norms that have the shape of "permissions" and they are represented as production rules.

3. Technological artifacts: The challenge is to build technological artifacts that facilitate and ensure the conscientious deployment of HOSS. One way of addressing this is to have an artifact for each modular component of the design workbench the components that are needed to assemble those modules. Again, for *uHelp* there is a specification language *SIMPLE* [7], that is interpreted by the *uHelp app*. An ambitious approach towards thorough implementations is to have *full platforms* that allow a translation form a specification to technological platform that implements that specification. The [2] volume discusses this line, and several frameworks for meta-modelling and implementation are available [1]. Another way to achieve this formal soundness is to start with an existing platform —*BrainKeeper*, *Amazon Turk*, *Ushahidi*— provide its formal counterpart and use ti to analyse applications of th platform.

4. Empirical study of HOSS: Complementing Topic 1, we find two further reasons to study working HOSS. One is to document compliance and failure of conscientious principles and recommendations, the other is to use the information that arises from their use as source data for socio-cognitive research.

5. Methodologies for conscientious design and deployment The challenge is to develop a precise conceptual framework to describe conscientious features and methodological guidelines that prescribe how to recognise and achieve the intended properties and behaviour in conscientious HOSS. We need to explore key values like fairness, trustworthiness, social empathy in principled terms (see [12,18]) so that we can speak properly of achieving engineering tasks like requirement elicitation or tooling conscientiously.

7 Peroration in four claims

First: The era of online social systems that on the surface seem to satisfy augmented social needs is *here to stay*. However, the rise of such systems has been so dramatic that *we simply do not know* what the effects will be either psychologically, sociologically,

culturally or politically.

Second: Some online social systems that involve human and artificial agency (HOSS) exhibit behaviours like hidden agency, inadequate stakeholder empowerment and lack of social empathy that may be problematic and deserve to be prevented or contended with in a sound manner.

Third: The challenge we face is to develop precise notions and the associated methodological guidelines and tools to design HOSS systems in a *conscientious* way that is *thorough*, *mindful* and *responsible*.

Fourth: This paper is a *call to arms* for such an initiative. Those of us working in the theory, design and implementation of agent-based systems, work in a field where there is an unharvested opportunity to apply our methods and tools in ways which could have impact far beyond that we might have imagined. It may mean a changing of the focus of our community and having to break away from our comfort zones describing idealised scenarios for agents, and in doing so we would need to be extremely humble about what we might achieve. But we should try, as the potential for sustained lasting impact for social and cultural good is potentially large.

The responsibility is substantial but the opportunity is ours.

Acknowledgements

The authors wish to acknowledge the support of SINTELNET (FET Open Coordinated Action FP7-ICT-2009-C Project No. 286370) in the writing of this paper. This research was partially supported by project MILESS (MINECO TIN2013-45039-P).

References

1. Huib Aldewereld, Olivier Boissier, Virginia Dignum, Pablo Noriega, and Julian Padget. *Social Coordination Frameworks for Social Technical Systems*. Number 30 in Law, Governance and Technology Series. Springer International Publishing, 2016.
2. Giulia Andrighetto, Guido Governatori, Pablo Noriega, and Leendert W. N. van der Torre, editors. *Normative Multi-Agent Systems*, volume 4 of *Dagstuhl Follow-Ups*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.
3. P. Brey. Values in technology and disclosive computer ethics. In L. Floridi, editor, *The Cambridge Handbook of Information and Computer Ethics*, pages 41 – 58. Cambridge University Press, Cambridge, 2010.
4. Cristiano Castelfranchi. InMind and OutMind; Societal Order Cognition and Self-Organization: The role of MAS. Invited talk for the IFAAMAS "Influential Paper Award". AAMAS 2013. Saint Paul, Minn. US. http://www.slideshare.net/sleeplessgreenideas/castelfranchi-aamas13-v2?ref=httpMay 2013.

5. Rob Christiaanse, Aditya Ghose, Pablo Noriega, and Munindar P. Singh. Characterizing artificial socio-cognitive technical systems. In Andreas Herzig and Emiliano Lorini, editors, *Proceedings of the European Conference on Social Intelligence (ECSI-2014), Barcelona, Spain, November 3-5, 2014.*, volume 1283 of *CEUR Workshop Proceedings*, pages 336–346. CEUR-WS.org, 2014.
6. David Collingridge. *The Social Control of Technology*. St. Martin's Press, London, 1980.
7. Dave de Jonge and Carles Sierra. Simple: a language for the specification of protocols, similar to natural language. In Murat Sensoy Pablo Noriega, editor, *The XIX International Workshop on Coordination, Organizations, Institutions and Norms in Multiagent Systems*, Istanbul,Turkey, May 2015.
8. Mark d'Inverno, Michael Luck, Pablo Noriega, Juan A. Rodriguez-Aguilar, and Carles Sierra. Communicating open systems. *Artificial Intelligence*, 186(0):38 – 94, 2012.
9. L. Floridi, editor. *The Onlife Manifesto: Being Human in a Hyperconnected Era*. Springer International Publishing, Cham, 2015.
10. B. Friedman, editor. *Human Values and the Design of Computer Technology*. Cambridge University Press, Cambridge, 1997.
11. J. R. Galliers. The positive role of conflicts in cooperative multi-agent systems. In Y. Demazeau and J.-P. Mueller, editors, *Decentralized AI: Proceedings of the First European Workshop on Modelling Autonomous Agents in a Multi-Agent World*. Elsevier, 1990.
12. Andrew J. I. Jones, Alexander Artikis, and Jeremy Pitt. The design of intelligent socio-technical systems. *Artif. Intell. Rev.*, 39(1):5–20, 2013.
13. C. P. Knobel and G. C. Bowker. Values in design. *Commun. ACM*, 54(7):26–28, 2011.
14. Andrew Koster, Jordi Madrenas, Nardine Osman, Marco Schorlemmer, Jordi Sabater-Mir, Carles Sierra, Dave de Jonge, Angela Fabregues, Josep Puyol-Gruart, and Pere García. u-help: supporting helpful communities with information technology. In *Proceedings of the First International Conference on Agreement Technologies (AT 2012)*, volume 918, pages 378–392, Dubrovnik, Croatia, 15/10/2012 2012.
15. Pablo Noriega, Julian Padget, Harko Verhagen, and Mark d'Inverno. The challenge of artificial socio-cognitive systems. In A. Ghose, N. Oren, P. Telang, and J. Thangarajah, editors, *Coordination, Organizations, Institutions, and Norms in Agent Systems X*, Lecture Notes in Computer Science 9372, pages 164–181. Springer, 2015.
16. N. Osman and C. Sierra. A roadmap for self-evolving communities. In A. Herzig and E. Lorini, editors, *Proceedings of the European Conference on Social Intelligence (ECSI-2014), Barcelona, Spain, November 3-5, 2014*, volume 1283 of *CEUR Workshop Proceedings*, pages 305–316. CEUR-WS.org, 2014.
17. Whitney Phillips. Loling at tragedy: Facebook trolls, memorial pages and resistance to grief online. *First Monday*, 16(12), 2011.
18. J. Pitt, D. Busquets, and S. Macbeth. Distributive justice for self-organised common-pool resource management. *ACM Trans. Auton. Adapt. Syst.*, 9(3):14, 2014.
19. John R. Searle. What is an institution? *Journal of Institutional Economics*, 1(01):1–22, 2005.
20. Eric Trist. The evolution of socio-technical systems. *Occasional paper, Ontario Ministry of Labour*, 2, 1981.

The Role of Values

Klara Pigmans, Huib Aldewereld, Virginia Dignum, and Neelke Doorn

Delft University of Technology, Delft, The Netherlands

Abstract. Decision-making processes involving multiple stakeholders can be rather cumbersome, turbulent and lengthy. We take as an example ongoing discussions in the Netherlands concerning the decision whether or not to flood pieces of land as a compensation for earlier lost ecological landscape. The stance of some stakeholders, upholding their individual interests, can slowdown or even block such processes. Recent research suggests that a focus on the values of the stakeholders could benefit those decision-making processes. However, the role of the values is not yet fully understood. To investigate the interaction between values, norms, and resulting actions in decision-making processes, we introduce a taxonomy to explore the relations between these concepts. The taxonomy presented in this paper is a first step towards a framework to model decision-making processes with the aim of understanding the role that values play in decision-making processes.

1 Introduction

Decision-making processes with multiple stakeholders can be complex, depending on stakeholders' behaviour [11,12]. For example, in the Netherlands, the decision about flooding the Hedwig polder has been a heated debate among the stakeholders. The decision to flood the polder of 299 hectare located in South-Western Netherlands, was taken already in 1977 to compensate for earlier lost ecological landscape. This decision has been both contested and supported ever since, by the different involved stakeholders, which include local residents, Dutch and various Belgium parliaments, environmental groups, farmers, and the European Commission. This is a classic example of how the stance of the stakeholders can slowdown or even block the decision-making process, and correspondingly the related (plans for) development.

To understand the development of such decision-making processes and the reason why some of them are turbulent or cumbersome, we need to explore the relation between the concepts involved in those processes. Research [6,9] suggests that values can play an important role in decision-making processes and that a value sensitive approach could therefore benefit such processes.

Moreover, at a closer look, it seems that it is not necessarily a value that influences the process. On the contrary, values are generally so vaguely defined that stakeholders all acknowledge their importance in abstract terms. It is rather the conception [10] that stakeholders have of this value

that can differ among the stakeholders and that influences their take on the process. E.g. justice is a value that is generally considered to be important and therefore supported. Yet, what justice entails, is a topic of debate.
In this paper we present a taxonomy to explore the relation between values, value conceptions, norms and the corresponding actions. By doing so, we take a first step towards the means to model these concepts in a decision-making context, which is needed to understand the way these concepts interact and how they influence the decision-making processes. The penultimate goal of this research is to explore and show what role values take in decision-making processes and whether a focus on the values and value conceptions provides a better means to solve difficult cases, as suggested by the earlier research in [6,9].
The remainder of this paper is structured as follows. In the next section we discuss the ideas behind the concepts, based on literature. In section 3 we describe and depict the collective structure of decision-making processes, and the taxonomy of the role of values in these processes. Section 4 discusses the context of this research by describing related work. In section 5 our conclusions and ideas for future work are presented.

2 Background

Before we can come to a taxonomy of values in decision-making processes, we first need to understand what the relevant concepts are and why these are taken into account.

2.1 Values

Values are defined in many different ways, e.g. as an enduring belief that a specific end-state is desirable over another [13], what a person or group considers important in life [7], or as guiding principles of what people consider important in life [2].
We assume that values can be considered to be more or less universal, like Schwartz and Rokeach state in their separate value surveys [2], but also like the values in decision making as stated by [1]. Justice, freedom, benevolence, and security are values that are broadly considered important in different cultures, organisations, and societies. The interpretation of these values is a different story, as explained in section 2.2.
In addition, ample research has been done on value typologies. The surveys of [14] resulted in 10 key value types, including power, hedonism, benevolence and security, describing relations between values. Earlier, [13] concentrated on the connection between values and behavior, distinguishing *terminal values* such as 'family security' and 'freedom', and *instrumental values* such as 'courage' and 'responsibility'. Since we are taking the decision-making process as our point of reference, the value hierarchy for management decisions [1] provides an interesting model as well. Bernthal distinguishes a business firm level, economic system level, society level, and an individual level. In multi-stakeholder decision-making processes in the public sector, these levels are very relevant: often

stakeholders are involved that are entrepreneurs or companies with business level values, including profits, survival, growth. Then if resources are involved, economic system values apply, such as allocation of resources, production and distribution of goods and services. The governmental authorities are likely to have societal values: culture, civilization, order and justice. Last, individuals will have values such as freedom, opportunity, self-realisation, and human-dignity.

2.2 Context and Value Conceptions

The difference between values and their interpretations is –in slightly different wording–, described by [10] as *contestable concepts* and *conceptions*. He describes contestable concepts as unitary and vague concepts, while their conceptions are contested since they are the arguments for how the concept should be interpreted in practice. Examples of contested concepts are liberty and social justice, which in this research we consider as values.
We assume that the context of a stakeholder or agent defines how a value is perceived. This context is the physical and social setting in which people live or in which something happens or develops. It includes the culture that the individual was educated or lives in, and the people and institutions with whom they interact[1].
This means that one agent can have multiple contexts, since it can e.g. work in a certain organisation, live in a certain CO_2-neutral community, and at the same time has fishing as a hobby, be member of a fishing community
These contexts influence the conceptions people have of values.

2.3 Vision and Collective Decision-Making Process

Since this research focuses on values in decision-making processes in particular, we include the vision and the collective decision-making process in our conceptualisation. The vision is expressed by an authority in long term documents or in vision reports in which the values of the authority are articulated in terms of vision, mission and plans. This vision represents the institutional objective that is set to realise the values, as also discussed in [5] as part of the abstract level. In order to accomplish this vision a collective decision-making process has to take place. In this process, the vision, different roles, and the norms of the agents are combined to come to a decision about which action to take.

2.4 Agents, Norms and Actions

For the definition of agents, norms and actions, we follow the vast body of research as presented in e.g. COIN and NorMAS, specifically, for this paper we use definition of agents as indicated by [8]. The decision-making process has several stakeholders, which are represented as agents. An agent can represent an individual stakeholder or a collective of stakeholders, e.g. farmers that unite their voice during the process.

[1] Definition from Wikipedia: $https://en.wikipedia.org/wiki/Social_environment$

3 Taxonomy of the Role of Values

The taxonomy that we present in this section has both an individual structure, describing the concepts that are relevant for the individual agents, as well as a collective structure representing the collective concepts of the decision-making process. We first describe the two structures separately, after which we connect them into the taxonomy. All is explained using an example.

3.1 The Collective Structure

The collective concepts in multi-stakeholder decision-making processes represent the commonalities in the process. The collective structure in itself seems rather straight forward, as depicted in figure 1.

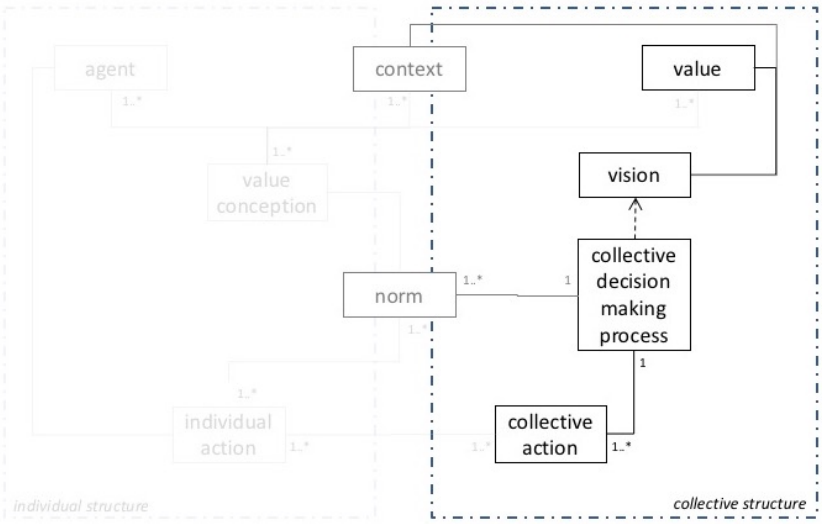


Fig. 1. Collective structure.

The collective decision-making process is initiated to realise the vision of authorities. This vision is derived from one or more values which are underlined by the involved authorities. The decision-making process leads to collective actions that will contribute to the realisation of the vision, and therefore the value.

We go over the structure step-by-step starting with the value. In this case we use the example of *water safety* as the underlying value.

The **vision** expresses how the value will be realised in terms of a 'collective objective', e.g. no floods should occur in the urban areas of the region. The vision is expressed in long term planning reports by the province and the municipality, including at least the value *water safety*,

but other values, such as *culture* could be expressed in the vision as well. For simplicity sake, we only focus on one value here.
In figure 1, the **collective decision-making process** follows from the vision.. The collective decision-making process does not take place at a single moment in time, but includes meetings, discussions, deliberations, one-to-one meetings, newsletters, informative events and compensation negotiations. In policy making, it often it takes decades to get to the point where a decision is actually agreed upon.
The **collective action** following from the collective decision-making process is in the end agreed upon by all agents. In the water safety example, the action could be to adjust the flow of the river that causes floods in the urban areas in the region, to evacuate an area or build a dike.

3.2 The Individual Structure

Because of the many inter-dependencies with the collective structure, the individual structure can not be depicted as a stand-alone separate structure, but we can still discuss the concepts themselves individually.

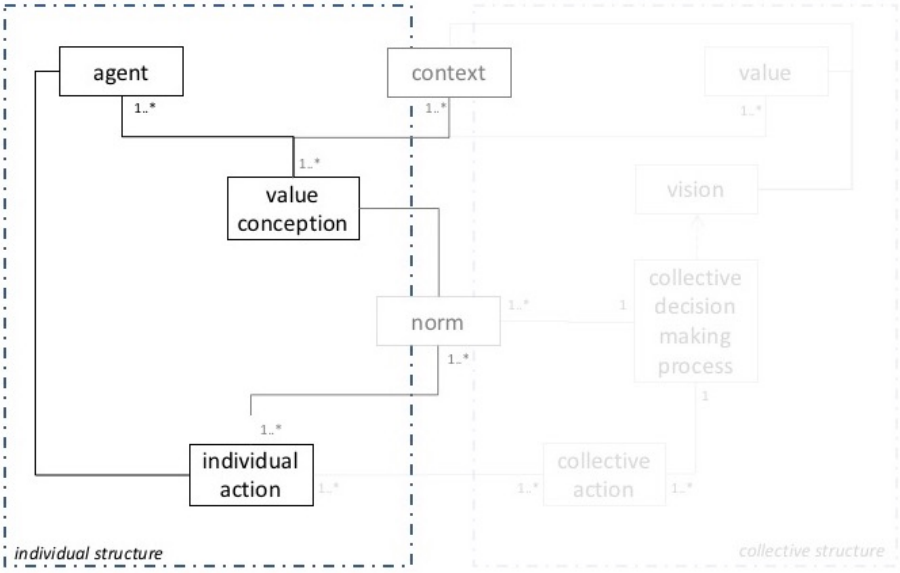


Fig. 2. The individual structure of decision-making processes.

A **value conception** is the interpretation of a value, so it has a direct relation to value, to the context that influences the value conception and the agent who has the value conception. With the value water safety, possible conceptions of water safety include risk prevention, flood defense,

flood mitigation, flood preparation, and flood recovery. Each of these conceptions contributes to the value water safety. In addition, an agent can have multiple conceptions: one agent can perceive flood defense and flood recovery combined as water safety.
The stakeholders that are involved are all represented as **agents**, this includes water authorities, municipality, province, inhabitants, agricultural entrepreneurs, property developers, and property owners.
The **individual actions** are taken by agents based on the norms they have. Individual actions could include lobby for/report on/organise a demonstration against/support that what is happening in the collective structure.

3.3 The structures combined in the taxonomy

The taxonomy of the role of values in multi-stakeholder decision-making processes is depicted in figure 3. The collective structure and the individual structure are related in multiple ways, including trough context and norms, which are part of both structures.

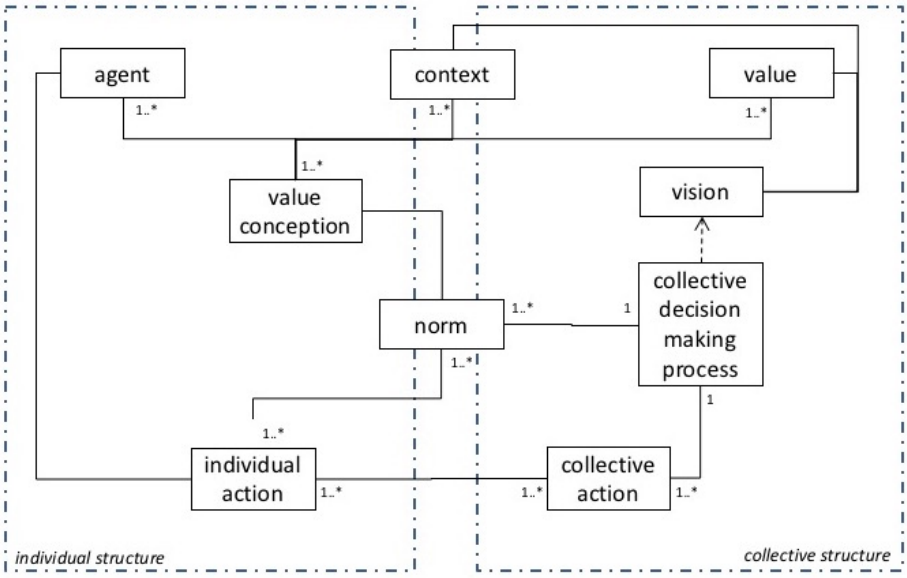


Fig. 3. Taxonomy of values, context, conceptions, norms, and actions in decision-making processes.

Value conceptions are influenced by the **context** of an agent. An authority representative who has been working for the water authority for years and lives in a different region, will be influenced by both the organisational culture and the geographical distance to the action that will be

taken; a local agricultural entrepreneur who has continued a large scale cattle farm after its parents could no longer run the business will in its turn be influenced by the history and inheritance of the company and by the geographical proximity to the actions to take.
Each value conception has a number of **norms**. The norm for a risk prevention conception could be that the chance that a flood occurs should always be less than 0,05% (fictional norm). The norm for the flood preparation conception could be 'the water should not exceed level x' (fictional norm).
Moreover, the vision follows from the value and the context. The vision 'no floods in urban areas' comes from the value water saftey in a context of water governance in a riverine region with both urban and countryside areas.
The socially or legally determined norms that agents have, e.g. the flood risk should be below a certain threshold, influence the collective decision making process.
Finally, the individual actions and the collective actions need to be aligned for the collective decision-making process to be succesfull. If the individual actions taken are 'demonstrating against the vision', then alignment with the collective action will be difficult.

4 Related work

In philosophical literature, e.g. [15], a direct relation between values and norms is indicated. Values, norms and design requirements are described as a value hierarchy, with values on top and design requirements at the bottom. There it is stated that values are specified by norms, which in their turn are specified by design requirements. The other way around, design requirements are in place for the sake of a norm, and a norm is in place for the sake of a value.
In the field of normative multi-agent systems, the use of values has been explored by [3], [4] and [5].
First, [3] describes the interaction between system norms –norms that are imposed on the agents by a system–, actions that are regulated by those norms, and personal values of the agents that are being promoted or demoted by those actions. While this is useful for the investigation into reasons why agents follow or violate norms, we believe that such a clear separation between the norms and values does not exist. Therefore, we express the need to further explore the way values and norms interact to determine collective and individual action.
Second, [4] argues that a value can be seen as a preference that can be discussed and debated. They describe norms to constitute a link between values and behavior, where norms serve this value. Their framework explores a connection between values, norms, goals and actions. In this research we want to take this one step further by exploring the role of these concepts in decision-making processes.
Third, the OMNI framework [5] discusses norms, values, context and social structures thoroughly, where each concept is located in a three by three matrix with three different levels and three dimensions. Yet,

values, agents, roles and actions are not discussed in terms of their direct relationship with each other, but rather in relation to the levels and the dimensions. To fully understand their role in decision-making processes we need to further explore these direct relations.

5 Conclusion and Future Work

Turbulent or cumbersome decision-making processes can slowdown or even block the plans for spatial development. Values are considered to play an important role in preventing or overcoming conflicts in such processes. In order to understand how values influence these processes, we discussed the relevant concepts and the relations between them. This resulted in a taxonomy with an individual structure and a collective structure. The individual structure of value conceptions, agents, and individual actions was then related to the collective structure, containing values, vision, collective decision-making process and collective action. Norms and context are concepts that are part of both structures. This taxonomy is the first step to explore and understand the concepts of decision-making processes.
So far, we did not take institutional aspects such as roles, norms and contexts into account. Further research is needed to expand the taxonomy with those aspects, including clear and detailed definitions on the attribute level. After expanding the taxonomy, the next step will be to formalise the concepts and relations, so that we can start modelling complex decision-making processes.

Acknowledgements

This work is part of the Values4Water project, subsidised by the research programme *Responsible Innovation*, which is (partly) financed by the Netherlands Organisation for Scientific Research (NWO) under Grant Number 313-99-316. The work of Neelke Doorn is supported by NWO under Grant Number 016-144-071.

References

1. W. Bernthal. Value perspectives in management decisions. *Journal of the academy of management*, 5(3):193–196, 1962.
2. A. Cheng and K. Fleischmann. Developing a meta-inventory of human values. *Proceedings of the American Society for Information Science and Technology*, 47(1):1–10, 2010.
3. K. Da Silva Figueiredo and V. Torres da Silva. Identifying conflicts between norms and values. In *Coordination, Organizations, Institutions, and Norms in Agent Systems IX*. Springer International Publishing., 2013.
4. F. Dechesne, G. Di Tosto, V. Dignum, and F. Dignum. No smoking here: values, norms and culture in multi-agent systems. *Articficial intelligence and law*, 21:79–107, 2013.

5. V. Dignum, J. Vazquez-Salceda, and F. Dignum. OMNI: Introducing social structure, norms and ontologies into agent organizations. In *Programming multi-agent systems*, pages 181–198. Springer Berlin Heidelberg, 2004.
6. N. Doorn. Governance experiments in water management: From interests to building blocks. *Science and Engineering Ethics*, pages DOI: 10.1007/s11948–015–9627–3, 2016.
7. B. Friedman, P. H. Kahn, and A. Borning. *The handbook of information and computer ethics*, chapter 4: Value sensitive design and information systems, pages 69–101. Wiley, 2008.
8. A. Ghorbani. *Structuring socio-technical complexities: modelling agent systems using insitutional analysis*. PhD thesis, Delft University of Technology, 2013.
9. L. Glenna. Value-laden technocratic management and environmental conflicts: The case of the new york city watershed controversy. *Science, Technology & Human Values*, 35(1):81–112, 2010.
10. M. Jacobs. Sustainable development as a contested concept. In A. Dobson, editor, *Fairness and futurity*. Oxford university press, 1999.
11. M. Kolkman, M. Kok, and A. van der Veen. Mental model mapping as a new tool to analyse the use of information in decion-making in integrated water management. *Physics and chemistry of the earth*, 30:317–332, 2005.
12. M. Reed. Stakeholder participation for environmental management: a literature review. *Biological conservation*, 141:2417–2431, 2008.
13. M. Rokeach. *The nature of human values*. New York: Free Press, 1973.
14. S. Schwartz. Are there universal aspects in the structure and contents of human values? *Journal of Social Issues*, 50(4):19–45, 1994.
15. I. Van de Poel. *Philosophy and engineering: reflections on practice, principles and process*, chapter 20: Translating Values into Design Requirements, pages 253–266. Springer Netherlands, 2013.

Representing human habits:
towards a habit support agent

Pietro Pasotti[1], M. Birna van Riemsdijk[2], Catholijn M. Jonker[3]

Abstract. Human behaviour is constrained by obligations on the one hand, by the routines and habits that constitute our *normal* behaviour on the other. In this paper, we present the core knowledge structures of *HabInt*, a Socially Adaptive Electronic Partner that supports its user in trying to adopt, break or maintain habitual behaviours. We argue that *HabInt*'s role is best conceived of as that of an extended mind of the user. Hence, we pose as requirements that *HabInt*'s representation of the relevant aspects of the user and her world should ideally correspond to that of the user herself, and use the same vocabulary. Furthermore, the knowledge structures of *HabInt* should be flexible and explicitly represent both its user's actual habitual behaviours and her desired habitual behaviours. This paper presents knowledge structures that satisfy the aforementioned requirements. We interleave their syntactic specification with a case study to show their intended usage as well as their expressive power.

1 Introduction

Man is a creature of habit. While people display a fascinating variety of behaviours even across relatively simple domains, it is also true that from day to day most people are quite fixed in their ways. Carrying out habitual activities is mostly unproblematic and even desirable ([29]). However at times unforeseen circumstances make our habitual choices unavailable or their outcomes undesirable. Other times we wish to adopt or break a habit, and both are difficult enterprises. While many of us normally have little or no difficulty in dealing with these challenges ([29]), the actual amount of nuisance is subjective. To some, even small disruptions of daily routines may cause anxiety and distress ([12, 18]), whereas for others, such as people suffering from depression, breaking habits can be beneficial ([24]). In this paper, we take the first steps in developing a concrete implementation of *HabInt*, a Socially Adaptive Electronic Partner ([31]) to support habit formation and breaking.

Our working definition of *habit* is the shared view among social psychologists in the tradition of Hull ([16]). They stress the Pavlovian nature of habits as goal-independent *learned associations between responses and features of performance contexts*.[4]

As argued in [32], a *habit* is not merely a *frequently performed behaviour*. A habit is best seen as a mental object *resulting from* repeatedly choosing the same behaviour when faced with the same choice in a stable context. A habit is thus an association between some fixed environmental (and temporal) cues and a learned response. The more a habitual behaviour consolidates, the more it acquires features including: a degree of *automaticity*; less need for attention/focus, so that it can be performed *concurrently* with other tasks; smaller emotional involvement; and finally a habit is *not goal-aware*: while typically consistent with one's goals, the goal it was originally directed at is no longer consciously pursued. (cfr. [35])

The two knowledge structures of *HabInt* that form the core of this paper are the Actual Behaviour Model (ABM) and the Desired Behaviour Model (DBM). The ABM encodes a set of overlapping chains of user activities and the ways in which they are typically performed. The DBM describes a 'contextually ideal' version of the ABM. The nodes of ABM are associated with information about the values they promote or demote, so that *HabInt* can keep track of the motivations behind the goals and construct a model of the user's preferences.

Section 2 describes the core aspects of *HabInt* architecture, and formulates the requirements for the knowledge structures representing actual and desired behaviour. The ABM is presented in § 3 and the DBM in § 4. In § 5 we give an overview of how *HabInt* can user the ABM and DBM information to monitor for various types of user anomalies. The related work is discussed in § 6. Finally, § 7, § 8 summarize our findings and point out directions for future work.

2 Habit support agent: what and how

This paper focuses on those structures of *HabInt* that represent the actual and desired behaviours of the user. The data they contain is accessed by a monitoring component which locates anomalies and hands them over to a module that determines what should the agent do to support the user, thereby closing the interaction loop (see Figure 1). By interacting with the user and monitoring her behaviour,

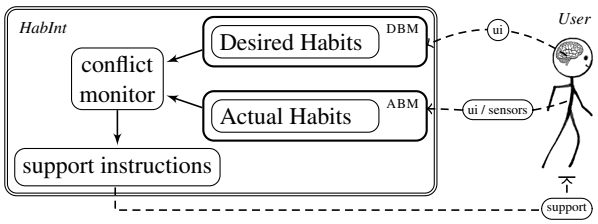


Figure 1. *HabInt*'s specific knowledge structures.

HabInt builds a model of the actual habits and typical activities of the user and a model of the user's desired habits. Desired habits are descriptions of behaviours which the user wants to turn into habits. These can be entirely new behaviours or changes to existing ones.

The monitoring module compares the user's desired and actual behaviour to detect conflicts: these are situations in which the user may need support. As a conflict is detected, a separate module that contains support instructions, previously provided by the user, is invoked. This module determines how should the agent intervene.

[1] TU-Delft,The Netherlands, email: P.Pasotti@tudelft.nl
[2] TU-Delft,The Netherlands, email: M.B.vanRiemsdijk@tudelft.nl
[3] TU-Delft,The Netherlands, email: c.m.jonker@tudelft.nl
[4] Cfr. [35] for an overview and further literature.

The ultimate goal of *HabInt* is to help the user achieve her goals and promote her values. In this sense, an implementation of a *HabInt* is best conceived of as part of the *extended mind* (see [9]) of its user. This means that it must be *trustworthy, reliable* and *accessible* (cfr. [9]) and so must be its knowledge. To make *HabInt* accessible and trustworthy, we must provide knowledge structures that are as transparent for the user as possible. The knowledge must be readily available for the user not only to use, but also to expand, contract and otherwise modify in a way that matches the way behaviours are discovered, explored, abandoned by people. To further enhance trust and reliance, we limit the agent's proactiveness to only those actions that are explicitly requested by the user. *Accessibility*, for one, means that the information/knowledge in the system must be easily accessible by the user. Consequently the *HabInt* has to store and manipulate its user model explicitly (unlike, for example, a neural network). For another thing, the user model should match the one the user has of herself as closely as possible, i.e. it should be a *shared mental model* (see [17]). Thus *HabInt* builds the vocabulary of goals, values, activities and actions from the user's wording. Summing up, *HabInt*'s knowledge structures should be:

adaptable: obtained by interacting with the user. This entails that they need to tolerate runtime updates and be built incrementally, while remaining meaningful at all intermediate stages of the construction process.

shared: correspond as much as possible to the user's conceptual structure and use the same vocabulary as the user does.[5]

explainable: *HabInt* needs to be able to carry out reasoning and explain the reasons that led to its current beliefs, in a dialogue referring to goals, values, and situational aspects ([30]). For example, *HabInt* should be able to model and then explain back to the user as requested which values are positively or negatively affected by some activities, to which goals activities contribute, and which values motivate which goals.

expressive: the structures need to accommodate uncertain, incomplete, and even inconsistent information. Finally, they must express the (context-dependent) behaviour enactment likelihood, for that is how *HabInt* can tell whether a behaviour is a habit or not, or whether it is becoming or ceasing to be one.

To show *HabInt*'s intended usage and the expressive power of its knowledge structures, we introduce a few snapshots of the life of a woman, Alice, as she interacts with *Hal*, her *HabInt*. Throughout the paper we will refer back to these scenarios and show how they are dealt with behind the scenes by *Hal*.

SCENARIOS: Alice and *Hal*

S1 Alice has a new job and would like to form a robust routine for travelling there. Also she would like to stop oversleeping. To help her with these issues Alice buys an *HabInt*, which she calls *Hal*. After booting it, Alice explains that she has two goals: first, to *'wake up'* and then to *'get to work'*. *Hal* asks what the options regarding the two goals are. It discovers that while there are a number of ways to get to the workplace, there is only one way of waking up, which requires remembering to set an alarm.

Alice explains to *Hal* that the main ways of getting to work are 1) by car, and 2) by bike. Furthermore, one can go by bike in two ways, 2.1) via the fast but risky Route A, or 2.2) via the safer, but longer Route B.

S2 Alice sometimes takes a cab to work. She feels no need for support in doing so, so when *Hal* reminds her to check the weather as she is leaving for work, she just says "well, actually today I'm going to work in some other way, so I won't need it. You don't need to worry about this." *HabInt* does not know how Alice is going to work that day.

S3 Alice now has the habit of setting the alarm every single day. However, exceptionally, on Mondays she forgets to set the alarm almost every other week (Probably this relates to her Sunday night's Vodka Tasting Club meetings).

S4 Alice tells *Hal* that of the two options to go to work by bike, she prefers the safe route (2.2) over the fast one (2.1). She explains that being fast is not as important to her as being safe.

S5 Alice asked *Hal* to help her grow the habit of going to work by bike. Years later, however, Alice decides to stop biking to work and go by car instead. Thus specific habitual behaviours part of her previous biking-to-work-routine are no longer necessary. She tells *Hal* the following: "(Instead of going by bike) now I'd like to go work by car", "I'll also need to stop taking the raincoat as I go to work."

S6 Alice long ago told *Hal* that she dislikes *'smoking'*, an action, because it demotes *'health'*, which she greatly values. Consequently, she has not smoked a single cigarette for 10 years now. However, one day *Hal* learns that Alice is smoking. After inquiring, *Hal* is told simply: "I *want* to start smoking."

3 The Actual Behaviour model (ABM)

There are habits regarding *what activities* we carry out daily; i.e. habits regarding, once something is done, what do we do *next* (`next`-*habits*). We model such activity patterns by capturing the sequential activation patterns of the goals that they purport to achieve.

Second, there are habits regarding *the way* in which we carry each activity out. We will call them `conc`-*habits*, for *concretisation habits*. The intuition is that just like the goal *get home by car* is intuitively more concrete that the goal *get home*, the activity of *driving home*, which achieves the former, is more concrete than the activity of *going home*, which achieves the latter. Achieving the former goal entails achieving the latter, but the converse does not hold. This *is-a-way-of* relation between goals is what we intend to capture with the notion of *concretisation*: we model habits regarding the way in which we do things by modelling the underlying goal concretisation patterns.

Finally, there are habits regarding *what actions* we perform as part of carrying out an activity (in a particular way): we call them *Action-habits*. For every activity, we represent the actions the user can perform when she tries to achieve its goal, and capture the likelihood that they are in fact performed.

In § 3.1 and § 3.2 we describe a knowledge representation language based on these three notions. Finally, exploiting our representation of the actions' consequences, we can express the values that they affect, and hence talk about the motivation and preferences that underlie behaviour choice and change. This is done in § 3.3.

The common basis of the language that the ABM is built upon is a language of alphanumeric strings. *HabInt* parses the User's messages at the level of propositional logic operators and treats the remaining uninterpreted strings as atoms. For example, "[the user is] not eating" becomes $\neg$*'eating'*. A propositional language over Strings[6] $\mathcal{L}_{\texttt{str}}$ is the basis of the knowledge structures we define next. A logical consequence relation is defined on formulae of $\mathcal{L}_{\texttt{str}}$ in the standard way. We use $\texttt{a}, \texttt{b}, \texttt{l}$ as variables ranging over $\mathcal{L}_{\texttt{str}}$.

3.1 Activities: what we do and how we do it

Abstracting away the temporal features for the sake of simplicity, an *Activity* is informally understood as *something which the user does to modify the current state of affairs*. Most of our daily activities

[5] I.e. if the user refers to its habit of 'brushing teeth after every meal', then that, literally, is the name *HabInt* stores.

[6] We capitalise technical terms, to avoid confusion with common concepts.

are carried out with a purpose, which we call a Goal. Our working definition of Goal is: *a declarative description of the state of affairs which the user would like to achieve by carrying out an Activity*.

HabInt's most fundamental knowledge structure represents the user's daily activities' underlying goals, and what goals are concretisations of what other goals. We call this knowledge the *Goal Base*.

The relation `conc` defines a branching structure of Goals that represents the way the user conceptualises her daily goals in terms of more concrete versions of themselves. This is captured by the binary relation `conc`. A special role is played by *toplevel Goals*, which are not a concretisation of any other Goal. In other terms, those for which the user sees no need to provide a higher goal. Examples for Alice include being awake early, having breakfast, getting to work and back home again. Toplevel Goals are then linked to one another by the relation `next`, forming a separate branching structure. This structure represents the user's potential Goal activation sequences: information about what she *might* do *after* doing something else.

Definition 1 (Goals and Goal Base) *The user's Goal Base is a triple* $\boldsymbol{G} := \langle G, \texttt{conc}, \texttt{next} \rangle$*, where:*

- $G \subseteq \mathcal{L}_{str}$ *is the current set of Goals of the user, with typical variables* g *and* g'.
- $\texttt{conc} : G \times G$ *is a* directed and acyclic concretisation relation*, such that* $\forall g, g' \in G : \texttt{conc}(g, g')$ *iff* g' *is a concretisation of* g.
- $\texttt{next} := \texttt{top}_{\boldsymbol{G}} \times \texttt{top}_{\boldsymbol{G}}$ *is a* directed, acyclic *relation such that* $\texttt{next}(g, g')$ *if once she has satisfied goal* g*, the user may try to satisfy (i.e. adopt)* g' *next.*

Furthermore, $\texttt{top}_{\boldsymbol{G}}$ *is the set of* toplevel goals *of* $\boldsymbol{G}$*:*

$$\texttt{top}_{\boldsymbol{G}} := \{g \in G \mid \forall g' \in G(\neg \texttt{conc}(g', g))\}$$

Note that the user can specify as many Goals as she likes, and can leave gaps and blanks. So, even if she habitually smokes at home, her *HabInt* may never know. It is up to the user to inform *HabInt* of alternative activities for the goals she mentioned to it, even if the user leaves these *under*specified. Hence *HabInt* must assume that unknown, additional alternatives always exist.

By monitoring and interacting with the user, *HabInt* learns and keeps track of the Activities that she carries out as she tries to achieve her goals, and of the sequences of actions that compose these Activities. All the actions *HabInt* is aware of are kept track of in the Action Base. The way we express Actions is standard practice:

Definition 2 (Actions and Action Base) $\mathcal{L}_{act}$ *is the language of actions, which are defined as formulae over* $\mathcal{L}_{str}$ *of the form*

$$\alpha \in \mathcal{L}_{act} := [\texttt{l: a} \rightarrowtail \texttt{b}]$$

where `l` *is the* name*,* `a` *the* precondition *and* `b` *the* postcondition *of the Action. The Action Base* $\boldsymbol{C} \subseteq \mathcal{L}_{act}$ *is the current set of Actions.*

By making the pre- and postcondition more detailed, the agent can represent each of the user's Actions in more or less detail, as well as specify the way in which they can be sequentially executed. We assume in what follows that *HabInt* has a planning module enabling it to reason about how to chain Actions together based on this (technicalities omitted due to lack of space).

Activities group up actions that can be performed as part of achieving some goal, and assign a name to the full bundle. If the *Act* field of an Activity is empty, that means that either performing that Activity is obvious enough to the user (i.e., she needs no support on that) or that she does not know yet. In that case, all an Activity does is associate a declarative goal with an informal (and meaningless to *HabInt*) description of a possible way to achieve it. All known Activities are stored in the Activity Base.

Definition 3 (Activities and Activity Base) *Given a set of Actions from the Action Base Act* $\subseteq \boldsymbol{C}$*, a goal* g *from the Goal Base* $\boldsymbol{G}$*, and a name* $\texttt{l} \in \mathcal{L}_{str}$*, an* Activity $\mathcal{A} \in \mathcal{L}_{uac}$ *is a tuple of the form:* $\langle \texttt{l}, g, Act \rangle$*.* `l` *is the name,* g *the goal, and Act the set of actions of the Activity. The* Activity Base $\boldsymbol{A} \subseteq \mathcal{L}_{uac}$ *is the current set of Activities.*

Through the Actions that compose them, Activities, too, can be made more or less fine-grained. Activities whose Goals are toplevel encode those activities that the user perceives as being self-justified or motivated by some of her values.

These Goal-Activity structures can be viewed as a variant of Goal-Plan Trees [27] where the `conc`-relation corresponds to OR-nodes, AND-nodes are left implicit, and distinct GPTs can be connected by the `next` relation.

[Behind the scenes of S1] *Hal* performs natural language analysis and determines that Alice's utterances mean the following: Alice wants support with two activities: going from home to work and waking up. The corresponding Goals are *'is awake'*, ①, and *'is at work'*, ②, respectively. Furthermore, *'go by bike'* and *'go by car'*, are names of activities whose corresponding declarative goals are *'is at work'* ∧ *'biked to work'*, ③, and *'is at work'* ∧ *'drove to work'*, ④. It has also recorded how going by bike/by car are ways of going to work, but going to work seems not to be a way to do something else, and is thus toplevel. The ABM is now as in Figure 2.

When Alice mentions how waking up requires having set the alarm, an Action α achieving ② (i.e. with ② as postcondition) is specified, which requires *'alarm set'* to be true. Formally, $\alpha = [$*'wake up'*: *'alarm set'* ➛ *'is awake'*$]$. Now $\mathcal{A}$ is $\langle$*'waking up'*, *'is awake'*, $\{\alpha\}\rangle$.

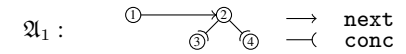


Figure 2. Hal's ABM of Alice.

In a nutshell, the construction process of the ABM is as follows: first, the user specifies a number of goals and whether each goal stands in a `conc` or `next` relation to some other known goal. Secondly, the user gives the names of activities that can achieve that goal, and, finally, she can describe the relevant actions that take part in carrying out each activity. In this way, the user determines what is an appropriate amount of specificity.

[Behind the scenes of S2] *Hal* learns that Alice does go to work, but neither by car nor bike. Therefore *Hal* records a new Activity $\mathcal{A}$, whose goal g_0 (a novel placeholder) is a concretisation of *'at work'*. $\mathcal{A}$ is named *'unknown alternative'*. Maybe one day Alice will tell *Hal* that she is going to work by *'take[ing] a cab'*. If so, *Hal* will update its knowledge structures.

The above shows that the knowledge structures are expressive, and explainable in that by manipulating directly the utterances (as strings) *HabInt* can maintain a model using the same vocabulary as the user. Furthermore, it is straightforward to define update operations such as splitting an Activity into two sub-Activities, adding/removing Actions, and splitting Actions into longer chains.

3.2 Habits: the way we normally do what we do

As we mentioned in § 1, a habit is not just a "frequent behaviour". However, frequency, automatism, ease of performance and other features of habitual behaviours are correlated. in particular frequency

can serve as a *predictor* for the other features and it can be derived from observing and communicating with the user ([35]). Therefore, we chose to detect habits through the underlying behaviours' enactment likelihoods. We describe a user's day as a sequence of toplevel goals (given by `next`). Each of those can then be concretised in different ways (as described by `conc`), and each goal can be assigned, via an Activity, a set of Actions that can be executed whilst achieving it. This is information about what the user is known to sometimes do: it defines the space of possible *behaviours*. Each one of these may in practice be enacted rarely or never, and both their content and their performance frequency can change over time. Consequently, we keep the representation of what the user knows she may do (`next`, `conc`, Activities) separate from the expectations regarding what she *will* do.

We have seen in § 1 that habits are cued by contexts. Hence we must keep track of those parts of the context that are believed by the user (or by some internal learning algorithm) to cue some behavioural response. We call them *Triggers*. Let $\mathbf{T} \subseteq \mathcal{L}_{\texttt{str}}$ be a *finite set of known Trigger*. Let τ range over $\mathbf{T}$.

However, reacting to Triggers is not automatic: even in the presence of a Trigger the cued behaviour may not follow. Then we must record, given the presence of a Trigger, the likelihood of the associated behaviour occurring. This is captured by `prob`. Formally, `prob` is a function of type $(\mathbf{T} \times G \times (G \cup \mathcal{L}_{\texttt{act}})) \rightarrowtail [0,1]$. Intuitively:

- $\texttt{prob}(g'|\tau, g) = x$, for toplevel goals g and g', means that *given* that g has been *just achieved* and that the Trigger τ holds, then the user adopts g' with likelihood x. If g' is not toplevel, then it means that *while* she tries to achieve g, and given Trigger τ the user is expected to *adopt* g' with likelihood $x \in [0,1]$.
- $\texttt{prob}(\alpha|\tau, g) = x$ means that if the goal g is adopted and τ holds, then the user is expected to execute action α with likelihood x.

Some behaviours' Triggers can be unknown, or so frequent to be irrelevant. In that case the Trigger is `true`.

If, given a Trigger, the enactment likelihood of some behaviour is above a certain threshold $t \in [0,1]$, *HabInt* infers that the behaviour is a *habit*. We call t the user's *habit threshold*, which is the performance likelihood above which the user feels confident in calling something a *habit*. Given existing research (e.g. [36]), it is reasonable to assume that $t > 0.5$. The return values of `prob` are estimated based on information from the user and/or sensor data (cf. [20]). Now we must keep in mind a key property of the notion of Goal we employ here: Goals that are concretisations of the same goal cannot be adopted concurrently. While this is not generally true for `next`-related Goals, here for simplicity we assume it is.[7] For example, after waking up, Alice can *go to work* or *go to the beach*, not both. Also, as she goes to work, Alice can go *'by bike'* or *'by car'*, not both. Therefore, no matter how `prob` is calculated, the likelihoods must sum up to one on all outgoing `next` paths and on all outgoing `conc` paths too. Actions are executed independently from one another, so there is no such constraint there.

The data structures we have defined so far allow us to express the types of habits described at the beginning of this section: `next-`, `conc-` and action-habits. These correspond to transitions between `next`-related goals, `conc`-related goals, and $\langle g, \alpha \rangle$ pairs respectively.

[7] So at any given moment the user can adopt at most one toplevel goal g, and an arbitrarily long `conc` chain of Goals with g at one end.

Definition 4 (Habits) *$\forall\, g, g' \in \boldsymbol{G}$, $\alpha \in \mathcal{L}_{\texttt{act}}$, $\tau \in \boldsymbol{T}$, and given a habit threshold $t \in [0,1]$, we define:*

$$\texttt{hab}(g'|\tau, g) \iff \begin{cases} \texttt{next}(g, g') \text{ and } \texttt{prob}(g'|\tau, g) > t \quad \textit{or} \\ \texttt{conc}(g, g') \text{ and } \texttt{prob}(g'|\tau, g) > t \end{cases}$$

$$\texttt{hab}(\alpha|\tau, g) \iff \texttt{prob}(\alpha|\tau, g) > t$$

Intuitively, if g' is toplevel, $\texttt{hab}(g'|\tau, g)$ means that the user habitually *adopts* g' given that g has been *achieved* immediately before, and that τ holds: a `next`-habit. If g' is not toplevel, $\texttt{hab}(g'|\tau, g)$ means that given that g is *adopted* and τ holds, the user habitually *adopts* g' *too*. In other words, as g' is a concretisation of g, the user *habitually tries to achieve g by achieving g'*: a `conc`-habit. $\texttt{hab}(\alpha|\tau, g)$, finally, means that the user habitually executes α *given that she has adopted* g and τ holds: an Action-habit.

[Behind the scenes of S3] Suppose that *Hal* knows that Alice's habit threshold t is 0.89. *Hal* then updates its ABM to reflect how her now-established habit of setting the alarm (the Action α) is endangered if the Trigger *'monday'* is present: while under no Trigger the behaviour has likelihood 0.9, when *'monday'* is the case it goes down to 0.6 (*HabInt* sets these values through interaction or monitoring).

$$\texttt{prob}(\alpha|\texttt{true}, \textit{'wake up'}) = 0.9$$

$$\texttt{prob}(\alpha|\textit{'monday'}, \textit{'wake up'}) = 0.6$$

3.3 Values: why we do what we do as we do it

Even though a *HabInt* having only the above structures can already be of use, in our opinion it needs to understand the motivations (based on values) for the choices the user makes to best support her. The user may need support in making satisfactory choices regarding her behaviour, which involves comparing competing Actions, based on their outcomes; competing Activities, based on the Actions they are associated with; and competing Goals, based on the Activities that they can be achieved by. For supporting the user, *HabInt* needs to understand and reason with the motives of the user's behaviour, and thus needs to know and understand her values. Paraphrasing [19],

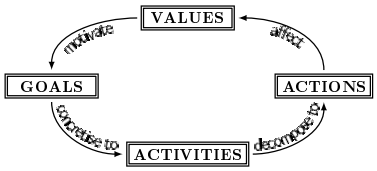


Figure 3. How values close the concretisation loop.

we understand *values* as *a hierarchy of desirable, abstract, cross-situational goals*. Ultimately, any activity is motivated by the pursuit of values. Still, all actions that we take as part of any activity end up affecting the same values (see Figure 3). So the user interface must be capable of value-based argumentation, and it is therefore natural to store also these knowledge structures in the unified world model we are describing here. As our *HabInt* is a *personal* support agent, here we assume that every user has her own (hierarchy of) values and hence we ignore their often-alleged universality ([23]). With the help of the user, *HabInt* learns what values she has, how important they are relative to each other, and what world features (literals from $\mathcal{L}_{\texttt{str}}$) can affect them. *HabInt* reasons about Values using value-based argumentation frameworks (cfr. [5]).

Definition 5 (Values and Value Base) *We define* $\boldsymbol{V} := \langle V, \lhd, \texttt{pro} \rangle$ *to be the Value Base of the user, where*

- $V \subseteq G$ *denotes the set of given Values of the user.*
- $\lhd \subseteq V \times V$ *is a preorder, such that* $\forall v, v' \in V : v \lhd v'$ *holds if* v is less important than v'.
- $\texttt{pro} := \mathcal{L}_{str} \times V \rightarrowtail \{\downarrow, \nmid, \uparrow\}$ *is an* injective *function encoding the way literals* a *from* $\mathcal{L}_{str}$ *promote (*$\uparrow$*), demote (*$\downarrow$*) or not affect (*$\nmid$*) the user's values.*

Note that the default return value of the function pro is $\nmid$: we assume that the user does not know or does not care, until she says otherwise.

When the user and her *HabInt* are reasoning about the best course of action to take, the postconditions of the actions involved play the fundamental role. Each postcondition expresses not only the goal its Action achieves but (in conjunctive normal form) a list of its effects. Exploiting this fact, *HabInt* can infer from the Value Base the way Actions first, then Activities affect Values.

As abstract goals of activities, values may well be unspecified and in the background.[8] But when it comes to evaluating the effects of the concrete Actions that together form an Activity, the importance and visibility of values become greater. Actions can be said to promote and demote values *by bringing about their postcondition* and, through the Actions that *habitually achieve them*, so can Activities. While Actions' outcomes are stable, habits dictate which Actions are executed when carrying out an Activity. Therefore, to determine what values are affected by an Activity, one must factor in habits.

With the Value Base, all parts of the ABM have been discussed.

Definition 6 (Actual Behaviour model (ABM)) *The* Actual Behaviour Model *is the tuple* $\mathfrak{A} := \langle \boldsymbol{V}, \boldsymbol{G}, \boldsymbol{A}, \boldsymbol{C}, \boldsymbol{T}, \texttt{prob}, t \rangle$*, where the elements are respectively, the Value Base, the Goal Base, the Activity Base, the Action Base, the set of Triggers, the conditional likelihood function, and the habit threshold.*

Given pro, which tells how $\mathcal{L}_{str}$ literals affect Values, we generalise it to $\texttt{pro}^*$, which also tells how Actions and Activities do.

Definition 7 (Promote) *Given an* ABM $\mathfrak{A}$*, the function* $\texttt{pro}^* := ((\mathcal{L}_{str} \cup \mathcal{L}_{act} \cup \mathcal{L}_{uac}) \times V) \rightarrowtail \{\uparrow, \downarrow, \nmid\}$ *is defined as follows:*

- *If* a *is a literal from* $\mathcal{L}_{str}$*, then* $\texttt{pro}^*(a, v) = \texttt{pro}(a, v)$.
- *If* $\varphi \in \mathcal{L}_{str}$*, then we require all the disjuncts to 'agree' on* v*:*

$$\texttt{pro}^*(\varphi \vee a, v) = \begin{cases} \texttt{pro}(a, v) & \text{if } \texttt{pro}^*(\varphi, v) = \texttt{pro}(a, v) \\ \nmid & \text{otherwise} \end{cases}$$

- *Let* α *be an action* $[1\texttt{:}\ a \blacktriangleright b]$*, and cnf*$(\alpha)$ *denote the* set of b's conjunctive normal form's conjuncts *(with* $\square \in \{\uparrow, \downarrow, \nmid\}$*). Given:*

$$C^{\alpha}_{\square v} := \{\varphi \in \mathit{cnf}(\alpha) \,:\, \texttt{pro}^*(\varphi, v) = \square\}$$
$$\texttt{pro}^*(\alpha, v) = \uparrow \quad \mathit{iff} \quad |C^{\alpha}_{\uparrow v}| > |C^{\alpha}_{\downarrow v}| \tag{1}$$

  similar to [33], we say that α *promotes a Value* v *(*$\texttt{pro}^*(\alpha, v) = \uparrow$*) if it brings about more* v*-promoting than* v*-demoting postcondition. The conditions for* $\texttt{pro}^*(\alpha, v) = \downarrow$ *or* $= \nmid$ *are very similar: change '*$>$*' to '*$<$*' and '*$=$*' in* (1) *respectively.*
- *Let* $\mathcal{A} = \langle 1, g, Act \rangle$*, and* $h(\mathcal{A}) := \{\alpha \in Act \mid \exists \tau, g : \texttt{hab}(\alpha | \tau, g)$ *holds in* $\mathfrak{A}\}$*; then*

$$D^{\mathcal{A}}_{\square v} := \{\alpha \in h(\mathcal{A}) \,:\, \texttt{pro}^*(\alpha, v) = \square\}$$
$$\texttt{pro}^*(\mathcal{A}, v) = \uparrow \quad \mathit{iff} \quad |D^{\mathcal{A}}_{\uparrow v}| > |D^{\mathcal{A}}_{\downarrow v}|$$

  $\texttt{pro}^*(\mathcal{A}, v) = \uparrow$ *means that the activity* $\mathcal{A}$ *promotes* v*. The conditions for demoting or not affecting* v *are again very similar.*

This is crucial for *HabInt* to represent inconsistencies between what the user does, or wishes to do, and his Values (cf. § 5 for an example).

[Behind the scenes of S4] *Hal* learns that Alice considers *'be safe'* (v_1) and *'be fast'* (v_2) as Values. Hence, it adds them to its previously empty Value Base, which now is $\mathbf{V} = \langle \{v_1, v_2\}, \varnothing, \varnothing \rangle$. Then it learns that biking through Route A (an Activity $\mathcal{A}$) promotes *'safety'*, but it does not know what specific postcondition of what Action involved in the Activity promotes it. Hence, first *Hal* adds a dummy Action $\alpha = [$*'something': 'something'* $\blacktriangleright$ *'*a_0*'*$]$ to $\mathcal{A}$ (where *'*a_0*'* is a new atom), and then adds to its Value Base the fact that $\texttt{pro}($*'*a_0*', 'safety'*$) = \uparrow$. Via the same process, it also records that Route B promotes *'be fast'*. From this, *Hal* can deduce that $\texttt{pro}^*(\mathcal{A},$ *'safety'*$)$. Finally, it learns: *'be fast'* $\lhd$ *'be safe'*. (Actually things are a bit more complicated, as promoting *'safety'* seems to be a property the Activity *always has*, according to Alice. Hence, by chaining post- and pre-conditions appropriately, *Hal* must ensure that α is presumed executed by the user *every time* the Activity of *'biking through route A'* is performed.)

4 The Desired Behaviour model (DBM)

People that are not quite satisfied with their actual behaviour may tell their *HabInt* what is bothering them. Only then, can they describe how they would like to be supported in changing it.

While the ABM of a user describes what the user does (and how she does it) in specific situations, the Desired Behaviour Model (DBM) describes a set of *Desired Habits* to the ABM that reflect how the user would like her ABM to become. The key intuition here is that if conforming to a desired behaviour were not an issue under any circumstance, the user would not mention it to *HabInt*. Therefore, *HabInt* treats each Desired Habit as *a support request*, which still does not convey any information about how the agent can in practice support the user. Later on, each Desired Habit can be linked to one or multiple ways in which the agent can support the user: for instance, instructions of when and how to produce a reminder, initiate a conversation, monitor some environmental variable, or ask what is going on. However, we do not discuss these in this paper.

In what follows, $\mathfrak{A}$ is an ABM, τ is a Trigger, g, g' are Goals, and α is an Action (all from $\mathfrak{A}$). We consider Desired Habits of three types:

next-Desired Habits are structures of the form $\langle \tau, g, g' \rangle$, where $\texttt{next}(g, g')$ is part of the Goal Base of $\mathfrak{A}$. This Desired Habit type formalizes the user's desires concerning her toplevel goal sequences. When she talks about what she should or would prefer to habitually do after doing something else, *HabInt* will formalise that as a next-Desired Habit.

conc-Desired Habits are structures of the form $\langle \tau, g, g' \rangle$, where $\texttt{conc}(g, g')$. This Desired Habit formalizes habit change desires concerning the way the user achieves some goal (i.e. her concretisation patterns). conc-Desired Habits formalise, for example, the user's desired habitual way of achieving some toplevel Goal.

Action-Desired Habits are structures of the form $\langle \tau, g, \alpha \rangle$, where there is some activity $\mathcal{A} = \langle 1, g, Act \rangle$ in $\mathfrak{A}$'s Activity Base with $\alpha \in Act$. If the user wishes to change the actions she habitually performs as part of carrying out some Activity, that will be formalised as an Action-Desired Habit.

In a similar fashion we introduce *undesired* behaviours or the habits which the user wants to drop. We call them *Undesired Habits*. They are also expressed in $\mathcal{L}_{\texttt{amd}}$ but stored in a different set, Undhab. Each Undesired Habit encodes the user's *desire to habitually not* enact a behaviour (in some way) or perform an action (given some

[8] Think about the habitual activity of going back home (after a day of work). The user can, but does not need to specify which values that macroscopic activity promotes.

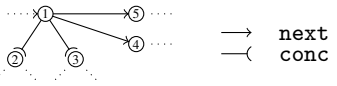


Figure 4. An example Goal structure. The Goal ① has two concretisations, ② and ③. Also, after achieving ①, the user can try to achieve either ④ or ⑤.

Trigger). The only constraint we impose is that Undhab and Dhab be *disjoint*, for obvious reasons.

Definition 8 (Desired Habits and Undesired Habits of the ABM) *Given an* ABM $\mathfrak{A} = \langle \boldsymbol{V}, \boldsymbol{G}, \boldsymbol{A}, \boldsymbol{C}, \boldsymbol{T}, \texttt{prob}, t \rangle$*, the set of* Desired Habits *is* Dhab $\subseteq \mathcal{L}_{dhab}$*, and the set of her* Undesired Habits *is* Undhab $\subseteq \mathcal{L}_{dhab}$*, where* $(g, g'$ *are Goals in* $\boldsymbol{G}$*,* $\tau \in \boldsymbol{T}$ *and* $\alpha \in \boldsymbol{C})$*:*

$$\mathcal{L}_{dhab} ::= \langle \tau, g, g' \rangle \mid \langle \tau, g, \alpha \rangle$$

With the difference between next- and conc-Desired Habits in mind, we can clarify their intended semantics by specifying the conditions under which they can be said to be complied with. A Desired Habit points out a behaviour which *should be habitual* under some trigger; hence a Desired Habit is complied with when that behaviour is indeed a habit (under that trigger). Similarly, a Undesired Habit is complied with when the corresponding behaviour is *not a habit*.

Definition 9 (Compliance) *Given a habit threshold* t *and an* ABM $\mathfrak{A}$*, we say that* $\mathfrak{A}$ *complies with*

$$\langle \tau, g, g' \rangle \in \texttt{Dhab} \quad \textit{iff} \quad \texttt{prob}(g'|\tau, g) > t \tag{2}$$
$$\langle \tau, g, \alpha \rangle \in \texttt{Dhab} \quad \textit{iff} \quad \texttt{prob}(\alpha|\tau, g) > t \tag{3}$$
$$\langle \tau, g, g' \rangle \in \texttt{Undhab} \quad \textit{iff} \quad \texttt{prob}(g'|\tau, g) < t \tag{4}$$
$$\langle \tau, g, \alpha \rangle \in \texttt{Undhab} \quad \textit{iff} \quad \texttt{prob}(\alpha|\tau, g) < t \tag{5}$$

Based on the known Triggers, *HabInt* keeps track of what behaviours the user wishes to change and stores them in its Dhab and Undhab. The Dhab, Undhab and ABM constitute the DBM.

Definition 10 (Desired Behaviour Model) *Given the* ABM $\mathfrak{A}$*, the Desired Habits* Dhab*, and the Undesired Habits* Undhab*, the* Desired Behaviour Model *is* $\langle \mathfrak{A}, \texttt{Dhab}, \texttt{Undhab} \rangle$.

[Behind the scenes of S5] Initially, Undhab is empty. However: Dhab $= \{\langle$true, *'at work'*, *'biked'*$\rangle\}$, because Alice originally wanted to form the habit of biking to work.

When Alice changes her mind, *Hal* firstly has to move $\langle$true, *'at work'*, *'biked'*$\rangle$ from Dhab to Undhab. Then, *Hal* formalizes Alice's desire to drive to work as the conc-Desired Habit: $\langle$*'rain'*, *'at work'*, *'drove'*$\rangle$ and *adds it to* Dhab. Now *Hal* knows:

$$\texttt{Dhab} = \{\langle \texttt{true}, \textit{'at work'}, \textit{'drove'} \rangle\} \tag{6}$$
$$\texttt{Undhab} = \{\langle \texttt{true}, \textit{'at work'}, \textit{'biked'} \rangle\} \tag{7}$$

Since Alice now also wants to drop the habit of "getting the raincoat" as she leaves for work (an Action $\alpha = [$*'get raincoat': 'at home'* ➛ *'has raincoat'*$]$), *Hal* has to further update Undhab to:

$$\texttt{Undhab} = \{\langle \texttt{true}, \textit{'at work'}, \textit{'biked'} \rangle, \langle \texttt{true}, \textit{'at work'}, \alpha \rangle\}$$

Other types of Desired Habits could in principle have been defined. For example, looking at Figure 4, one may wish to express Dhab(true, ②, ⑤). It could be read as requesting to form a habit of "instead of doing ① by means of ②, stop doing ① altogether and start doing ⑤ instead". But this is rather convoluted, and we see little added value. Rarely we say things like: "if you see me go to work *by bike*, remind me I should stay home instead". For similar reasons also the other possible Dhab-types require more far-fetched interpretations. So, we will not discuss them further.

5 Violation, anomaly, and inconsistency monitor

The structures we have described so far capture (un)desired habits, one-off behaviours, existing habits, and also the user's values, and *all can be at odds with one another*. Hence many types of conflict can be expressed in their language. Here we describe three: the most crucial ones to monitor for habit support. Namely we show that, given the DBM and ABM, *HabInt* can monitor whether an actual behaviour is *anomalous*, *inconsistent* with the user's value-based preferences or whether it *violates* an existing Desired Habit. The examples point out how *HabInt*'s monitoring module can check the user's ABM and DBM for such conflicts (all examples refer to Figure 4).

behavioural anomaly: when the user does something unusual (or in an unusual way). For example, when hab$(\tau, ①, ⑤)$, but the agent believes that the user is now doing ④ instead of ⑤.

When a behavioural anomaly is detected, *HabInt* can e.g. be instructed to investigate, remind the user of her habitual behaviour, or alert a supervisor. The ABM knowledge alone is sufficient for expressing this anomaly. Both ABM and DBM are needed, on the other hand, to express the following state of *violation*: when a Desired Habit is not a habit (or vice versa, when an Undesired one is).

$$\langle \tau, g, g' \rangle \in \texttt{Dhab} \wedge \langle \tau, g, g'' \rangle \notin \texttt{Dhab} \wedge \langle \tau, g, g'' \rangle \in \texttt{hab}$$

undesired behaviour: when the user does something (in a way) she declared she does not want to (or should not). For example, if $\langle \tau, ①, ④ \rangle \in \texttt{Dhab} \wedge \langle \tau, ①, ⑤ \rangle \notin \texttt{Dhab}$, but the agent believes that the user habitually does ⑤ after ①, when τ.

When *undesired behaviour* is detected, this means that the user is doing something she declared she wanted support in *not doing* (or vice versa). Many kinds of support can be associated with violations of this type. For example the user may ask to be reminded of the values she invoked when she set the Desired Habit she is about to violate or to talk once more about the consequences of her behaviour. The same holds for one-off behaviours in place of habits.

Furthermore, using the notions introduced in § 3.3, *HabInt* can reason about which Values are affected by an Activity and know, for example, if an Activity $\mathcal{A}$ for the goal g demotes the user's most important values: $\forall v(\nexists v'(v \lhd v') \Rightarrow \texttt{pro}^*(\mathcal{A}, v) = \downarrow)$

If the user mentions that she is carrying out $\mathcal{A}$, or $\langle \tau, g', g \rangle \in$ Dhab, then her *HabInt* will detect a *value inconsistency*:

value inconsistency: when the user's Actions, Activities, or (Un)Desired Habits are not in line with her preferences. For example, when an Action demotes an important Value.

[Behind the scenes of S6] When *Hal* perceives Alice smoking, its *behavioural anomaly* handling would instruct it to ask: "what is going on?". But at the same time, *Hal* thought that Alice disliked smoking, "because smoking demotes *'health'*", so this also categorises as an *undesired behaviour* (i.e. she might be falling into old bad habits) and has to be dealt with differently. So *Hal* asks instead: "is everything all right?" When Alice tells *Hal*: "I *want* to start smoking. Every day after lunch, for a start.", then *Hal* will have to handle a *value inconsistency*: either *'health'* is not *that* important (any more), or maybe the user has forgotten the values behind her previous choice.

For *HabInt*, anomalies, violations and inconsistencies mean either that the user is in trouble, or that its information is outdated. If an undesired behaviour violation is detected, then a sought-for habit change process may not be going smoothly, and the agent can e.g.

deliver a warning, as previously instructed by the user. The *value inconsistency* type of anomaly can be a symptom of inconsistencies in the user's motivation/intention/action structure, or irrational behaviour. To find out which one it is, *HabInt* can be instructed to initiate communication with the user.

6 Related work

Research on human-computer interaction has explored many ways in which technology can be used to aid behaviour change (e.g. [25]) and support habit formation and breaking. In contrast with these approaches, our focus is not on the psychological aspects of behaviour change in a specific domain and how to support this through technology. Rather, our conception of *HabInt* is as an extended mind of the user: we focus on developing generic knowledge structures that allow a *HabInt* to represent and construct user habits in a way that corresponds to the her conception of her activities.

The field of Activity Recognition has developed machine learning approaches to deduce what an observed human being is doing (and her behavioural patterns too), based on raw sensor data. For examples and further references, see [25, 11, 20]. These techniques will be used in the monitoring component, to update the `prob` function and automatically determine what the user is doing. This reduces the amount of information we need to get directly from the user. Research such as [8, 10] on (often neural network-like) learners that mimic the acquisition and monitoring of routine sequential action in humans is related, but does not satisfy most of the requirements of § 2 due to its different purpose. Our knowledge structures are at a higher level of abstraction, and capture the relations between activities as well as their motivations to enable user support on the basis of these higher-level concepts. However, the `prob`-based transition system is inspired to Markov Models [13].

Another area has investigated agents that form habits and routines of their own (see for example [2, 15]). Instead, our interest is in an agent that *supports* a human user in dealing with her habits. Knowledge structures oriented towards habit-learning agents need not be shared or explainable, and consequently the models are not explicit.

The challenge of developing support agents capable of dynamically interacting with humans in complex environments is not new (e.g., [4, 6, 37]). We share with them the general vision of a support agent, but the domain of support for dealing with individual habits was still unexplored. Secondly, while existing support agents in this tradition build on the notion of an agent whose primary goal is to take over some of the tasks a human has, the *HabInt* we propose has no such purpose. As a consequence, we face some different issues. For example, the issue of Adjustable Autonomy, as in [22], disappears, because *HabInt* does not take over human tasks or responsibilities but simply automates some of them *when requested*. On the other hand, the question of how to time the interventions remains.

In the area of multi-agent systems, knowledge structures for the representation of goals and actions have been extensively studied (e.g. [7]). In this paper we show how such structures can be used as a basis for the representation of habits. In future work we will investigate to what extent BDI languages can be used as a basis for implementing *HabInt*. A difference between *HabInt* and BDI agents is that agents *execute* plans themselves while *HabInt* represents a *user's* goals and activities in order to support the user in executing them. Thus a core challenge towards *HabInt*'s development will be to study how to these knowledge structures can be constructed in interaction with the user, and to develop further notions and reasoning techniques for habit support on their basis.

7 Discussion

The literature agrees on habits being no longer consciously goal-oriented: *awareness* of the goal has been lost in the process of habit formation and is no longer an explicit motive, but at most a latent justification. While in other situations the motive needs to be present in order to motivate action, habitual behaviours often lose sight of the motives as soon as they are no longer needed.[9] Clearly, turning carefully deliberated-upon choices into habits is a value-driven endeavour of its own, since in so doing we free up time and effort-consuming deliberations by crystallising them into automatic cue-response mechanisms (e.g. [21]). But by obscuring the values the behaviours' goals used to promote or demote, the process of habit formation risks making us blind to better choices and pitfalls alike. *HabInt* can help to counter this phenomenon by recording explicit representations of the values that are promoted or demoted by certain behaviours. The habit-formation process weakens awareness of how actions affect values and how values motivate goals (cfr. Figure 3) *HabInt* can help to close the loop, so that the values that are the motivation and purpose of habitual behaviour can be made visible again, revealing perhaps its normative aspects.

The dictionary definition of the word *norm* ([1]) includes: ***a)*** *A principle of right action binding upon the members of a group and serving to guide, control, or regulate proper and acceptable behaviour* ***b)*** *A widespread or usual practice, procedure, or custom.*

This paper focuses on habits, which fall under the second meaning of *norm*. We find it interesting that habits often conflict with norms of the first kind (e.g. see [28]), abiding by which often requires conscious effort and self-control. This is at odds with the absent-minded, automatic way we carry out our daily routines. We believe that the common ground of both meanings of *norm* can be found in *values*.

We also remark that the link between the two kinds of norms is even stronger than it may initially seem when observed under the perspective we have outlined in this paper. *HabInt* has a model of what *is* actually the case, the ABM, and a model of what *should be* the case, the DBM, which is the user's own idea of a better self (one that wakes up in time, brushes her teeth, etc...). This idea echoes the normative-descriptive dichotomy of economists and psychologists ([26, 3]) on which deontic logics can be built, as shown e.g. in [14].

8 Conclusion and future work

While performing habitual behaviours is characteristically easy, forming and breaking habits can be difficult. Our aim is to develop a *HabInt* agent that is able to support humans in such efforts. We conceive of such an agent as an extended mind of the user. In this paper we have presented the foundations for developing this agent by outlining the vision and requirements, as well as providing knowledge structures for performing the necessary reasoning and support tasks. A case study illustrates the envisaged use of the framework.

The knowledge structures model habits on the basis of a representation of goals, activities and actions relevant to the user. The paper shows how these concepts can be linked to personal values which is essential for helping the user to choose desired behaviours that are in line with her underlying motivations. We have put forward the notion of a desired behaviour model as the basis for supporting a user in modifying habits. We have proposed several types of non-compliance based on the actual and desired behaviour models which can be used by the *HabInt* to monitor whether the user's actual behavior is in line

[9] This is a simplification: see [34] for a more complete account.

with her desired behavior. In this way, *HabInt* will be able to determine when and how to give the user timely advice.

A key feature of *HabInt* is that it adheres strictly to the user's vocabulary for expressing goals, activities, actions and values, and that the fine-grainedness of the concretisation relation and of the actions involved in the user's activities are tailored to the user's needs. If a reminder to "go by bike" is enough for the user to know what to do, then no additional information is stored by *HabInt*.

In future work we intend to investigate the resemblance to David Lewis' *perfect worlds* semantics for deontic logic. We would like to study whether deontic logic techniques would allow the type of reasoning needed to differentiate between actual behaviours, already formulated desired behaviours, and tentative attempts of the user to formulate her actual or desired behaviours.

The current model does not address the temporal dimension of habits. As timely interventions are crucial, the temporal aspects will be addressed in a future paper. Furthermore, here we defined `prob` as a probability function, whereas we would like *HabInt* to reason qualitatively, using notions such as 'always, often, at least biweekly...'. This challenge is to be addressed in future work, as will the other components of the agent architecture, the implementation and the verification of the validity, scalability and robustness of the approach.

ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

References

[1] http://www.merriam-webster.com/dictionary/. [accessed 15-05-2016].
[2] Philip E. Agre, 'The dynamic structure of everyday life', Technical report, MIT, (1988).
[3] David E. Bell, Howard Raiffa, and Amos Tversky, eds. *Decision Making: Descriptive, Normative, and Prescriptive Interactions*. Cambridge University Press, 1988.
[4] V. Bellotti, B. Dalal, N. Good, P. Flynn, D.G. Bobrow, and N. Ducheneaut, 'What a to-do: Studies of task management towards the design of a personal task list manager', in *Proceedings of CHI'04*, pp. 735–742, (2004).
[5] Trevor J.M. Bench-Capon, 'Persuasion in practical argument using value-based argumentation frameworks', *Journal of Logic and Computation*, **13**(3), (2003).
[6] P. Berry, K. Conley, M. Gervasio, B. Peintner, T. Uribe, and N. Yorke-Smith, 'Deploying a personalized time management agent', in *Proceedings of AAMAS'06*, pp. 1564–1571, (2006).
[7] Rafael H. Bordini, Mehdi Dastani, Jürgen Dix, and Amal El Fallah Seghrouchni, *Multi-Agent Programming: Languages, Tools and Applications*, Springer, Berlin, 2009.
[8] Matthew M. Botvinick and David C. Plaut, 'Doing without schema hierarchies: A recurrent connectionist approach to normal and impaired routine sequential action', *Psychological Review*, (2004).
[9] Andy Clark and David Chalmers, 'The extended mind', *Analysis*, **58**(1), 7–19, (1998).
[10] Richard P. Cooper, Nicolas Ruh, and Denis Mareschal, 'The goal circuit model: A hierarchical multi-route model of the acquisition and control of routine sequential action in humans', *Cognitive Science: A Multidisciplinary Journal*, (2013).
[11] Thi V. Duong, Hung H. Bui, Dinh Q. Phung, and Svetha Venkatesh, 'Activity recognition and abnormality detection with the switching hidden semi-markov model', in *CVPR 2005 : Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 838–845, (2005).
[12] Kerry Fairbrother and James Warn, 'Workplace dimensions, stress and job satisfaction', *Journal of Managerial Psychology*, **18**(1), 8–21, (2003).
[13] Shai Fine, Yoram Singer, and Naftali Tishby, 'The hierarchical hidden markov model: Analysis and applications', *Machine Learning*, (1998).
[14] Holly S. Goldman, 'David Lewis' semantics for deontic logic', *Mind*, **86**(342), 242–248, (1977).
[15] Henry H. Hexmoor, *Representing and Learning Routine Activities*, Ph.D. dissertation, New York State University, 1995.
[16] Clark L. Hull, *Principles of behavior: an introduction to behavior theory*, Appleton-Century, 1943.
[17] Catholijn M. Jonker, M.Birna van Riemsdijk, and Bas Vermeulen, 'Shared mental models - a conceptual analysis', in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, ed., van der Hoek, (2010).
[18] *Model of Human Occupation: Theory and Application*, ed., Gary Kielhofner, Lippincott Williams & Wilkins, 2008.
[19] Ariel Knafo and Shalom H. Shwartz, *Cultural transmission: Psychological, developmental, social, and methodological aspects*, chapter Accounting for parent-child value congruence: Theoretical considerations and empirical evidence., 240–268, Culture and psychology, Cambridge University Press, 2009.
[20] Óscar D. Lara and Miguel A. Labrador, 'A survey on human activity recognition using wearable sensors', *IEEE Communications Surveys and Tutorials*, **15**(3), 1192–1209, (2013).
[21] Eva Lindbladh and Carl H. Lyttkens, 'Habit versus choice: the process of decision-making in health-related behaviour', *Social Science & Medicine*, **55**(3), 451–465, (August 2002).
[22] David V. Pynadath and Milind Tambe, *Socially Intelligent Agents*, volume 3 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, chapter Electric Elves: Adjustable Autonomy in Real-World Multi-Agent Environments, 101–108, Springer, 2002.
[23] Meg J. Rohan, 'A rose by any name? the value construct', *Personality and Social Psychology Review*, **4**(3), 255–277, (2000,).
[24] Neil S. Jacobson, Christopher R. Martell, and Sona Dimidjian, 'Behavioral activation treatment for depression: Returning to contextual roots', *Clinical Psychology: Science and Practice*, **8**(3), 255–270, (September 2001).
[25] Katarzyna Stawarz, Anna L. Cox, and Ann Blandford, 'Beyond self-tracking and reminders: Designing smartphone apps that support habit formation', in *CHI '15: Conference on Human Factors in Computing Systems, Seoul, Republic of Korea, April 18 - 23, 2015.*, pp. 2653 – 2662, (2015).
[26] Carroll U. Stephens and Jon M. Shepard, *Wiley Encyclopedia of Management*, chapter Normative/Descriptive.
[27] John Thangarajah, *Managing the Concurrent Execution of Goals in Intelligent Agents*, Ph.D. dissertation, Royal Melbourne Institute of Technology, 2005.
[28] David Trafimow, 'Habit as both a direct cause of intention to use a condom and as a moderator of the attitude-intention and subjective norm-intention relations.', *Psychology and Health*, **15**, 383–393, (2000).
[29] Frank Trentmann, *Time, Consumption and Everyday Life: Practice, Materiality and Culture*, chapter Disruption is Normal: Blackouts, Breakdowns and the Elasticity of Everyday Life, Berg, 2009.
[30] M. van Lent, W. Fisher, and M. Mancuso, 'An explainable artificial intelligence system for small-unit tactical behavior', in *Proc. of the Sixteenth Conference on Innovative Applications of Artificial Intelligence*, (2004).
[31] M.Birna van Riemsdijk, Catholijn M. Jonker, and Victor Lesser, 'Creating socially adaptive electronic partners', in *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015)*, (2015).
[32] Bas Verplanken, 'Beyond frequency: Habit as a mental construct', *British Journal of Social Psychology*, **45**, 639–656, (2006).
[33] Wietske Visser, Koen V. Hindricks, and Catholijn M. Jonker, 'Argumentation-based qualitative preference modelling with incomplete and uncertain information', *Group Decision and Negotiation*, (1), 99–127, (2012).
[34] Wendy Wood and David T. Neal, 'A new look at habits and the habit-goal interface', *Psychological Review*, **114**(4), 843– 863, (2007).
[35] Wendy Wood and Dennis Rünger, 'Psychology of habit', *Annual Review of Psychology*, (2015).
[36] Wendy Wood, Leona Tam, and Melissa Witt G., 'Changing circumstances, disrupting habits', *Journal of Personality and Social Psychology*, (2005).
[37] Neil Yorke-Smith, Shahin Saadati, Karen L. Myers, and David N. Morley, 'The design of a proactive personal agent for task management', *International Journal on Artificial Intelligence Tools*, **21**(1), (2012).

"How Did They Know?" — Model-checking for Analysis of Information Leakage in Social Networks

Louise A. Dennis[1], Marija Slavkovik[2], and Michael Fisher[1]

[1] Department of Computer Science, University of Liverpool
[2] Department of Information Science and Media Studies, University of Bergen

Abstract. We examine the use of model-checking in the analysis of information leakage in social networks. We take previous work on the formal analysis of digital crowds and show how a variation on the formalism can naturally model the interaction of people and groups of followers in intersecting social networks. We then show how probabilistic models of the forwarding and reposting behaviour of individuals can be used to analyse the risk that information will leak to unwanted parties. We illustrate our approach by analysing several simple examples.

1 Introduction

Can we use formal verification to check whether the privacy settings for accessing posted content in social media are effective? In this work we make the first steps towards answering this question in the positive.

The proliferation of social network services has made it possible for vast amounts of contributed content to be shared online by users who simultaneously are members of more than one social network service (SNS). Consider, for simplicity, one SNS user; let us call him Bob. Most social network services allow for various privacy settings to be specified, which should allow Bob to control who can access or, further propagate, the content he contributes. We say "should allow control" instead of "does allow control" because, in reality, it is not Bob's privacy settings that ultimately determine accessibility to his shared content, but the combination of the privacy settings of Bob and the privacy settings of all of the users to whom Bob has allowed access to his shared content, *i.e.*, Bob's *followers*. In the same vein let us call Bob's *followees* all the users who have allowed access to their shared content to Bob. What is worse with respect to Bob's control over the privacy of his shared content, is that many of his followers may be users of more than one SNS, with automated interfacing set to synchronise their activities among all the mediums either because one social network allows direct linkage with the API of another (e.g., Livejournal[3] allows posts to be automatically reposted as a link to Facebook[4]) or via third party synchronisation services such as IFTTT[5] and Zapier[6] which allow users to create customised rules to link their SNS accounts to each

[3] `livejournal.com`
[4] `facebook.com`
[5] `ifttt.com`
[6] `zapier.com`

other (and often to additional services and devices such as home automation tools, calendars, alerts and emails). It is thus very difficult for Bob to track *information leakage* – information that Bob shares with his followers, but reach other agents who are not directly authorised to share it. We give a very simple example of information leakage.

Let Bob and his friend Cathy both be members of social network service SN1. Cathy and Bob are within each others networks on SN1, meaning they are both each other's followers and followees. In turn Bob's boss, Jim, is neither a follower nor a followee of Bob. Bob regularly posts content on SN1 and has chosen to make his content visible only to his followers, believing that his boss cannot access them. Bob makes really sure of this, he checks Cathy's followers and makes sure Jim is not among them. However Cathy and Jim are within each others networks on SN2 and Cathy automatically synchronises her posts between these two SNSs. Bob, having a hard day, complains about his boss on SN1. Cathy, sympathising with Bob acknowledges Bob's message thus making it visible to her followers on SN1, but due to her content synchronisation with SN2, Bob's message becomes also visible to Cathy's followers on SN2. As a result Jim finds out what Bob really thinks of him and rescinds his planned promotion.

It is not simple for one user such as Bob to keep track of all possible combinations of privacy settings within his network and their ultimate effect on content accessibility. Therefore we propose that this task of checking the effective content visibility, *i.e.*, that no information leakage has occurred, should be automated. As a possible means to accomplish such automation, we propose formal verification. Our ambitions aim is to make it feasible for social network services to regularly model-check [4] user settings to ensure that the content privacy settings are effective and efficient, although we are aware that this is a very hard theoretical and engineering problem.

Formal verification is the process of establishing, typically via techniques based on formal logic, that a designed system has its intended properties. Such approaches have become widespread, enabling deep and (semi) automated formal analysis of both software and hardware systems so providing greater clarity concerning reliability and correctness. While logical proof techniques can be used, it is exhaustive state-space exploration, in the form of *model-checking* [4], that is the predominant approach. As we wish to formally model SNSs, our aim here is to utilise formal verification tools to automatically verify their behaviour. In particular, we wish to establish formal properties concerning information leakage using automatic *model-checking* systems.

Consequently we begin, in §2 and §3 by considering the general class of systems and a specific formal model for these based on similar work for namely digital crowds [18] Indeed, the formal model here provides a simplification of that in [18] in that agents have much more limited capabilities. We then consider how model-checking can be used to analyse information leakage properties within this framework. This we do in §4, utilising the PRISM probabilistic model-checker [10]. Finally, in §5, we provide concluding remarks, incorporating both related and future work.

2 System Representation

A *rational* agent is an agent that is capable of obtaining information about her environment, including other agents, and using this information to select actions in order to

achieve her goals [20]. A multi-agent system (MAS) is a system of agents that share the same environment and can cooperate or compete within it, as well coordinate their actions. A system of social network services (SNSs) and their users is not a "traditional" MAS, foremost because the networks are not considered to be agents. We propose that since the SNS does obtain information about the users it hosts, and adapts its services and information to the particular needs of specific users, it can be modelled as a rational agent. We use the catch-all phrase "social agent" to refer to both SNSs and their users. We now discuss how to represent a social agent, so that we can formally analyse her properties.

A rational agent can be represented by representing her mental attitudes, in particular her dynamic, informational and motivation aspects. This is exemplified by the popular BDI paradigm for representing agents via mental attitudes [16, 15]. "BDI" denotes *Beliefs*, *Desires*, and *Intentions*. In terms of the analysis of information leakage we are primarily interested in the informational aspects of rational agency and so in what follows we will ignore the issue of an agent's desires and intentions[7].

As flexible and powerful as the BDI paradigm is, it is not completely suited for representing social agents since the mental attitudes of these agents, particularly if they are a SNS, are not available or they may not be visible. *E.g.*, a SNS may not have access to what Bob truly believes about his boss, only to what Bob has posted about his boss. Bob can know who Cathy's followers are on the SNS they share, but not on the SNSs they do not have in common. For reasons such as these, work in [18] introduces a new mental state, the communicational attitudes to describe the information about herself an agent shares with the world; $M^{\uparrow i}\varphi$ is used[8] to describe that the modelled agent has communicated φ to i, while $M^{\downarrow i}\varphi$ is used to describe that the modelled agent has received communication φ from agent i. An agent can be modelled by only using communicational attitudes, when nothing of the private beliefs or goals of the agent is known. The agent representation in [18] builds upon formal agent organisational structures introduced in [8] and further studied in [7, 9]. An extended agent, as given in [8], is one for which in addition to the specification of the agent's mental attitudes, two further sets are added, *content* and *context*, allowing for both simple agents and a system of agents to be represented using the same model. An extended agent, as defined in [7, 9], can further include an agent's specification that is visible, or accessible, to the agent's content or context respectively. This paradigm of extended agents is particularly suitable for modelling the visibility of posted content. We thus arrive at our model of a social agent, we use the content to represent the followers of a user and the context to represent the user's followees.

The model of a social agent is given in Fig. 1. The mental attitudes of the social agent are private and it is not necessary to include any information in this agent part in order to specify a social agent. The information the agent shares is accessible to the agents that are her followers. The followers also have access to information about who else follows the modelled social agent. Information received from a followee is naturally

[7] Though note that these could be included.

[8] In [18], the formulas $M^{\uparrow i}\varphi$ and $M^{\downarrow i}\varphi$ have also subscripts that denote the nature of the communication, *i.e.*, whether it expresses a question, a statement, or an order, but we here only use statements and thus omit subscripts.

accessible to the agent who posted that information. If an agent A_1 is followed by an agent A_2, then A_2 can know who else A_1 follows.

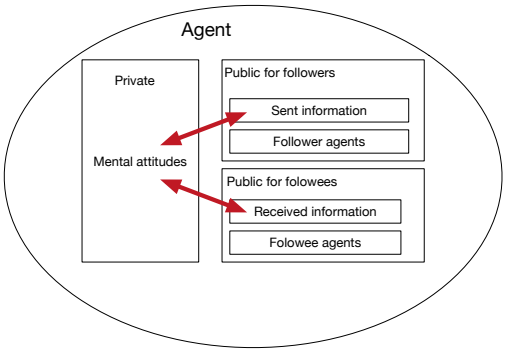


Fig. 1: Basic structure of an extended agent.

Using this social agent structure, we can construct a model for the simple information leakage example outlined in §1. This model is given on Fig. 2.

3 Formal System Specification

The systems we need to specify are the SNS and their users. We represent both networks and users as extended agents using a simplification of the extended agent representation given in [18]. In [18], additional modalities were used to express language abilities as well as the type of the message that the agent sends or receives, linguistic structures that we do not have need for here.

Let Agt be a set of unique agent identifiers, let $Prop$ be a set of atomic propositions and constants, and $Pred$ be a set of a first-order predicates of arbitrary arity. We begin by defining a language $\mathcal{L}_p$ to be a set of grounded first order logic formulas without function symbols, namely the set of all φ_p such that

$$\varphi_p ::= p \mid \neg\varphi_p \mid \varphi_p \wedge \varphi_p \mid P(x_1, \ldots, x_m)$$

where $p \in Prop$, $P \in Pred$ and $x_1, \ldots, x_m \in Agt$.

Depending on the specific needs for a specification, different BDI operators can be used but, for demonstrating our specification approach, we use only the modal operator B which denotes the agent's informational attitudes. $\mathcal{L}_{BDI}$ is then the set of all formulas φ such that

$$\varphi ::= \varphi_p \mid \neg\varphi \mid \varphi \wedge \varphi \mid B\varphi_p$$

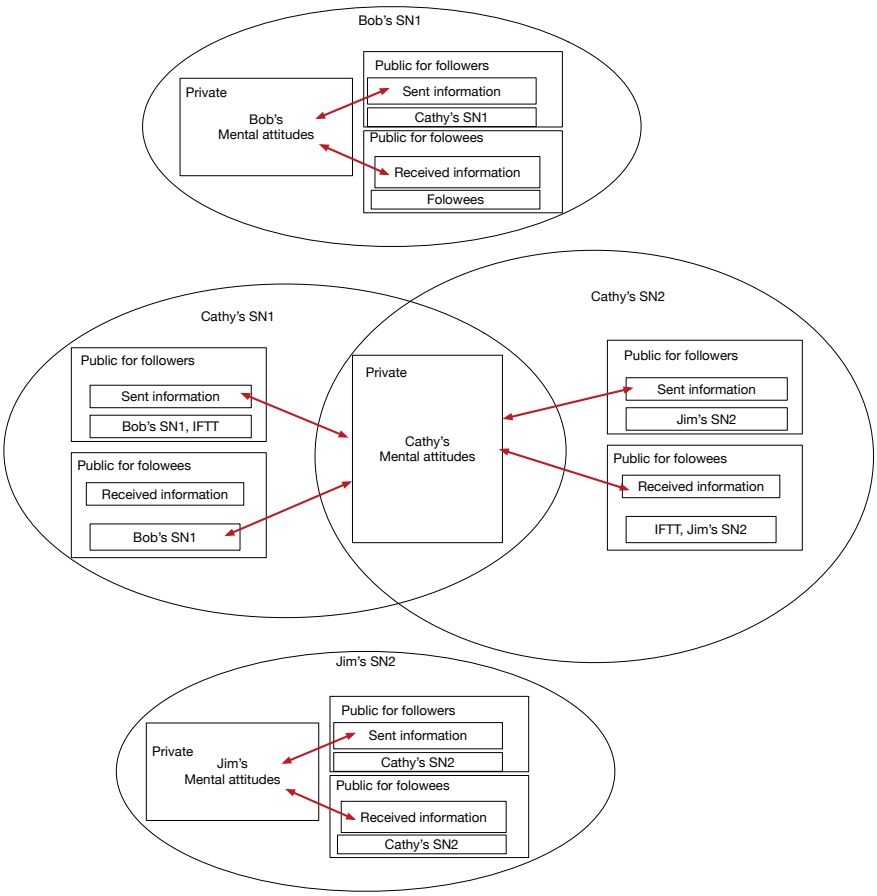


Fig. 2: A system of social agents

where $\varphi_p \in \mathcal{L}_p$.

Finally, we define the language for specifying communication among agents, $\mathcal{L}_M$. For this language we add operators to indicate the sending and receiving of messages and probabilities. The language $\mathcal{L}_M$ is the set of all formulas θ such that

$$\theta ::= \varphi \mid M^{\downarrow j}\varphi \mid M^{\uparrow j}\varphi \mid \neg\theta \mid \theta \wedge \theta$$

where $i, j \in Agt$ and $\varphi \in \mathcal{L}_{BDI}$. In [18], temporal information can be included in message formulas but we ignore that possibility here.

The messages are sent to an agent j, however either the *context* set CX or the *content* set CN as a whole can be the target of message broadcast (in the general model, both are agents). We use the shorthand[9]

$$M^{\uparrow CN}\varphi \equiv \bigwedge_{j \in CN} M^{\uparrow j}\varphi, \qquad M^{\uparrow CX}\varphi \equiv \bigwedge_{j \in CX} M^{\uparrow j}\varphi.$$

The language $\mathcal{L}_{BDI}$ restricts the nesting of modal operators, while $\mathcal{L}_M$ forbids the use of BDI operators outside of the scope of a message operator and does not allow nesting of M operators. Nested messages express meta communication, allowing agents to communicate about what was communicated to them or by them. However, such nesting is not meaningful in our work here.

We can now give the following definition of an agent.

Definition 1. *Let Agt be a set of unique agent identifiers. An agent is a tuple $\langle ID, Bel, Com, CN, CX\rangle$, where $ID \in Agt$ is a unique agent identifier, $Bel \subset \mathcal{L}_p$ is the set of beliefs the agent holds about the world, $Com \subset \mathcal{L}_M$ is the set of messages the agent has received and sent, $CN \subset \mathcal{P}(Agt\backslash\{ID\})$ is the set of agents contained and lastly $CX \subset \mathcal{P}(Agt\backslash\{ID\})$ is the set of agents in which the agent is contained, i.e., its set of contexts. The set Bel is consistent and simplified.*

Given an agent $i \in Agt$, an agent specification is a set $SPEC(i) \subset \mathcal{L}_M$, where $B\varphi$ is true iff $\varphi \in Bel$, $cn(j)$ is true iff $j \in CN$, $cx(j)$ is true iff $j \in CX$ and $M^{\downarrow\uparrow i}\varphi$ is true if $M^{\downarrow\uparrow i}\varphi \in Com$.

Lastly when specifying the behaviour of a system we combine probabilistic and temporal operators.

$$\varphi ::= \mathtt{P}^{=n}\theta \mid \theta \mid \varphi \mathbf{U} \varphi \mid \bigcirc\varphi \mid \Diamond\varphi$$

where $\theta \in \mathcal{L}_M$, $0 \leqslant n \leqslant 1$ and $\varphi \in \mathcal{L}_{BDI}$. In the intuitive interpretation of our probabilistic operator: $P^{=n}\theta$ means that there is a probability of n that θ is true. For our temporal logic operators $p\mathbf{U}q$ means that p is continuously true up until the point when q becomes true; $\bigcirc r$ means that r is true in the next moment in time; while $\Diamond s$ means that s will be true at some moment in the future. Note that temporal operators can not be nested within the probabilistic operator which, therefore, refers only to the probability of messages being sent or formulas in $\mathcal{L}_{BDI}$ becoming true.

Finally, we assume, via (1), that if a message is sent then it will eventually be received. This is a property of communication among agents that should hold in the environment, for communication to be meaningful.

$$\exists i, M^{\uparrow j}\varphi \in SPEC(i) \Rightarrow \exists j, \Diamond M^{\downarrow i}\varphi \in SPEC(j) \tag{1}$$

Note that we do not develop an axiomatisation for $\mathcal{L}_M$ and do not intend to prove soundness for this language, because we aim ultimately to use it to create specifications for model checking, where soundness is not necessary. The above, together with

[9] **Note:** We define the messages with individual agents, not sets as in [8, 7, 9], because a message can be broadcast to many agents, but it can be sent from one agent, otherwise the sender is unknown, which cannot happen here — if your contexts sends you a message it is from exactly one context.

standard modal and temporal logic semantic structures [19], provides a formal basis for describing agents and SNSs, communication and, hence, behaviour.

In order to consider communication among social networks, let us define the concept of *reachability* between two agents i and j. The agent i can reach agent j if, and only if, a message sent from i is eventually forwarded to j, under the assumption that the relevant contexts relay messages from one of their content agents to the rest of the content. Of particular interest in the analysis of information leakage are *relaying contexts*. Intuitively, a relying context is an agent which broadcasts to all its content agents all messages received from one of his content agents.

Definition 2. *Let i be an agent s.t. $CN(i) \neq \varnothing$. Agent $k \in CX(i)$ is a* relaying context*, and $REL(k)$ is true, when all the messages sent to k are sent on to all of the content agents of k:*

$$((CN(i) \vee CX(i)) \wedge M^{\downarrow i}\varphi) \rightarrow M^{\uparrow CN}\varphi) \in SPEC(k)$$

To show that information leakage to agent j does not happen to content posted by agent i we need to show that $SPEC(i)$ satisfies property (2):

$$\neg\Diamond(M^{\uparrow CN}\varphi \wedge \neg CN(j)) \rightarrow M^{\downarrow j}\varphi)) \tag{2}$$

Recall that CN are the followers of i, while CX are her followees. The property (2) states that it is not possible that what is posted to followers of i can be received by j who is not among i's followers.

Upon this basic framework we will now consider formal verification of key properties. To explain this, we will work through a relatively simple series of examples, showing the properties that can be formally established via model-checking.

4 Model Checking Information Leakage

PRISM [10] is a probabilistic symbolic model-checker in continuous development since 1999, primarily at the Universities of Birmingham and Oxford. Typically a model of a program (or in our case a network of agents) is supplied to PRISM in the form of a probabilistic automaton. This can then be exhaustively checked against a property written in PRISM's own probabilistic property specification language, which subsumes several well-known probabilistic logics including PCTL, probabilistic LTL, CTL, and PCTL*. PRISM has been used to formally verify a variety of systems in which reliability and uncertainty play a role, including communication protocols, cryptographic protocols and biological systems [14]. In this paper we use PRISM version `4.1.beta2`.

PRISM is an attractive option for modelling agents and social networks in our formalism since its probabilistic aspects allow us to reason not only about which messages are definitely sent and received, but also about the chance, or risk, that information leakage may occur.

We use a simple set of examples in order to illustrate our approach.

4.1 Basic Scenario

Alice, Bob, and Charlie share two social networks, SN1 and SN2. Alice is a follower of Bob on SN1 but Charlie is not. Charlie is a follower of Bob on SN2 but Alice is not. We treat all three agents, Alice, Bob and Charlie as *modules* in PRISM. Following our formalism we also treat the followers of Bob on the two networks as agents and so also as PRISM modules. The followers of Bob on SN1 and SN2 are both 'relaying' contexts as defined in Definition 2 – i.e. all information from one content member is is automatically transmitted to all other content members.

The syntax of prism commands is `[?label] guard -> prob_1:update_1 + ...+ prob_n:update_n` where `label` is an optional keyword used for synchronisation, `guard` is a logical formula over the values of global and local variables, `prob_1` to `prob_n` are probabilities which sum to 1 and `update_1` to `update_n` specify changes to the global and local variables.

We modelled our scenario as a *Discrete Time Markov Chain* in PRISM. Therefore '`->`' indicates a transition from one discrete time step to another. Synchronisation labels force commands in several modules to make a transitions at the same time.

We show the model for followers of Bob on SN1, `SN1Bob`, in Fig.3. In this model

```
module SN1Bob
sn1bob_relays_message: bool init false;

[bobmessagetosn1] bob_sent_message_to_sn1 = true ->
                          1.0:(sn1bob_relays_message' = true);
[sn1bobmessage] sn1bob_relays_message = true ->
                          1.0:(sn1bob_relays_message' = false);

endmodule
```

Fig. 3: A PRISM model of Bob's followees on SN1.

`bob_sent_message_to_sn1` is a variable in the Bob module that is true if Bob has sent a message to SN1. `sn1bob_relays_message` is a variable in `SN1Bob` that is true if SN1 relays a message from bob to all his followees on SN1. `SN1Bob` contains two PRISM commands, both with synchronisation labels. The first specifies that if Bob has sent a message to SN1 then, with a probability of 1.0, sn1 will relay the message. This transition is synchronised with commands in other modules labelled `bobmessagetosn1` (specifically it synchronises with a command in the Bob module that sends the message). The second specifies that if sn1 relays a message then a synchronised transition will take place after which this variable is set to false (pending receipt of a new message from Bob).

To represent the receipt of messages by Bob's followers we use the synchronisation label `sn1bobmessage`. All the commands with this label in all modules make transitions together. In practice this means all Bob's followers receive a message in the same time step. So, for instance, in the representation of Alice in the model, when SN1 relays Bob's message she, with probability 1.0, has a message.

```
[sn1bobmessage] sn1bob_relays_message = true &
                                1.0:(alice_has_message' = true);
```

If there were a second agent, Debbie say, among Bob's SN1 followers then Debbie would contain a similar command.

```
[sn1bobmessage] sn1bob_relays_message = true &
                                1.0:(debbie_has_message' = true);
```

Taken together the synchronised commands in the content agents and the relaying command in `SN1Bob` ensure that SN1Bob meets the specification of a relaying context.

4.2 Example 1

In our first, and simplest, example Alice, Bob and Charlie are the only relevant actors on each network. Bob posts a message to SN1. With the simple model and probabilities PRISM tells us that there is a probability of 1 that eventually Alice will receive the message[10]:

$$P^{=1}\Diamond M_{tell}^{\downarrow sn1bob}\ message \in SPEC(alice) \tag{3}$$

This is expressed as `P>=1 [F(alice_has_message = true)]` in PRISM's property specification language.

We can also prove that there is probability of zero that Charlie will eventually know the message, since the message was relayed only to Bob's followers on SN1 and not to those on SN2.

$$P^{=0}\Diamond M_{tell}^{\downarrow sn1bob}\ message \in SPEC(charlie) \tag{4}$$

4.3 Example 2

We now expand our example to consider the addition of a synchronisation agent, SYNC. Bob has set SYNC up so that when he posts a message to SN1 it is forwarded to the SN2 *as if it was Bob doing so*. We use a *global variable* `sync_sends_as_bob` to represent that sync can send a message as if it were Bob. When this variable is true then the Bob module sends the message to SN2 using the command

```
[] sync_sends_as_bob = true ->
        1.0: (bob_sent_message_to_sn2' = true) &
             (sync_sends_as_bob' = false);
```

The synchronisation agent is shown in Fig.4.
So, on receipt of a message from Bob by the first network, the SYNC agent forwards it to SN2 *as if it was Bob doing so*. Under these circumstances we can use PRISM to show that the probability that eventually Charlie receives the message is 1.

[10] We use the notation $P^{=n}$ to indicate that there is a probability of n that something will occur.

```
module SYNC
sync_has_message: bool init false;

[sn1bobmessage] sn1bob_relays_message = true &
                              sync_has_message = false ->
                    1.0:(sync_has_message' = true);

[] sync_has_message = true ->
                    1.0: (sync_has_message' = false) &
                         (sync_sends_as_bob' = true);
endmodule
```

Fig. 4: PRISM model of a simple synchronisation service

4.4 Example 3

Let us now remove the synchronisation agent and consider the possibility that Bob's followers on SN1 may forward the message to their followers. Assume both Alice and Debbie follow Bob and that Charlie follows both Alice and Debbie. With both Alice and Debbie there is a possibility of 0.1 that they may forward a message to their own followers.

$$\forall j \in \{alice, debbie\}, \forall i, M_{tell}^{\downarrow i}\varphi \in SPEC(j) \Rightarrow P^{=0.1} M_{tell}^{\uparrow SN1}\varphi \tag{5}$$

The PRISM model for Debbie's behaviour is shown in Fig.5 (Alice's module is identical except for variable names and labels). We also add new synchronisation commands to Charlie's model to indicate a receipt of messages from Alice or Debbie's SN1.

```
module Debbie
debbie_has_message: bool init false;
debbie_sent_message_to_sn1: bool init false;

[] debbie_has_message = true ->
                        0.9:(debbie_has_message' = false)
                        + 0.1:(debbie_has_message' = false) &
                              (debbie_sent_message_to_sn1' = true);

[sn1bobmessage] sn1bob_relays_message = true ->
                        1.0:(debbie_has_message' = true);
[debbiemessagetosn1] debbie_sent_message_to_sn1 = true ->
                        1.0:(debbie_sent_message_to_sn1' = false);

endmodule
```

Fig. 5: PRISM model for Debbie

In this network PRISM tells us there is a probability of 0.19 that Charlie will eventually receive the message having had it forwarded to him by either Alice or Debbie (or by both of them).

4.5 Example 4

Suppose at the same time that Bob sends his message he requests that it not be reposted. We view this request as the establishment of a norm and assume this further modifies the chance that Alice or Debbie will forward the message to 0.01. We represent this by modifying the behaviour of agents when they have a message as show in figure 6:

```
[] debbie_has_message = true & do_not_repost_norm = false ->
                      0.9:(debbie_has_message' = false)
                      + 0.1:(debbie_has_message' = false) &
                            (debbie_sent_message_to_sn1' = true);
[] debbie_has_message = true & do_not_repost_norm = true ->
                      0.99:(debbie_has_message' = false)
                      + 0.01:(debbie_has_message' = false) &
                             (debbie_sent_message_to_sn1' = true);
```

Fig. 6: PRISM command showing Debbie's behaviour when a norm is in place

Under these circumstances, PRISM tells us that the probability of Charlie receiving drops to 0.0199.

4.6 Example 5

Lastly we combine our various scenarios as follows: Bob is followed by Alice and Debbie on SN1 and by Charlie on SN2. Debbie and Alice are followed by Charlie on SN1. Debbie has a synchronisation agent set up on SN2 to forward her message automatically to SN1. Debbie is not followed by Charlie on SN2. If Bob asks that his message *not* be forwarded to Charlie then both Alice and Debbie have a 0.01 probability of reposting the message to SN1. However there is a 0.09 probability that Debbie will forward the message to SN2 since Charlie does not follow her there, forgetting that she has a synchronisation agent set up. In these circumstance the probability that Charlie receives the message is 0.109, either because Alice or Debbie has forwarded it directly to SN1, or because Debbie forwarded it to SN2 and then SYNC reposted it to SN1.

4.7 Results Summary

We summarise the results of our examples in the table below, in each case showing the probability, $P^{=?}$ that Alice, Charlie, Debbie or sync eventually receive Bob's message.

	Example				
	1	2	3	4	5
$P^{=?}\Diamond M^{\downarrow}_{tell}\varphi \in SPEC(alice)$	1	1	1	1	1
$P^{=?}\Diamond M^{\downarrow}_{tell}\varphi \in SPEC(charlie)$	0	1	0.19	0.0199	0.109
$P^{=?}\Diamond M^{\downarrow}_{tell}\varphi \in SPEC(debbie)$	n/a	n/a	1	1	1
$P^{=?}\Diamond M^{\downarrow}_{tell}\varphi \in SPEC(sync)$	n/a	1	n/a	n/a	0.09

5 Discussion

The analyses of information leakage that we have presented assumes that it is possible to gain some information about the composition of interlinked social networks in order to construct a model for analysis. In particular we assume that we can model the probability with which a user will forward messages; that we can gather information about the followers of users on different social networks (and identify users across social networks); and that we can tell when a user is using a synchronisation agent. We will briefly discuss each of these assumptions.

How likely is a user to forward a message? A decision made by an individual user over whether or not to repost a message to their own followers on a Social Network is obviously highly dependent upon the user, the content of the message, and external factors such as the time of day. However some work already exists in modelling the chances that a message becomes disseminated within a social networks [13] so it reasonable to assume that realistic probabilities could be generated to assess both the risk of messages in general, and of some specific message being forwarded within a network. Adding in assumptions about normative behaviour clearly makes such modelling harder however work also exists in modelling the norms of behaviour on social networking sites [3].

Can we gather information about a user's followers on different social networks and identify users across social networks. While some social networks make the list of a user's followers public, many do not and this obviously presents considerable difficulty in modelling the intersection of these networks. Moreover, for practical reasons the depth of exploration — i.e. the number of forwards — will need to be limited for search reasons. However, it would not be unreasonable to assume a model in which once a message has been forwarded n times it can count as having "gone viral" and the information therein has irrevocably leaked. We have not considered this possibility here. Typically forwarding of messages happens primarily within the network where the message was generated. In this instance the network itself could choose to offer information leakage analysis from its vantage point of access to all follower groups.

How can we tell if a user is using a synchronisation agent? The main danger of information leakage between networks arises when a user is employing a synchronisation agent. While it is generally easy to tell if a person you follow on a social network is using an agent to repost to that network from some other network, it is considerably harder to tell if they have a synchronisation agent that posts from the network you share to one that you don't. It may be that the existence of such agents for other users will need to

be modelled as part of user behaviour. However it is easy to obtain information about synchronisation agents for a the user wishing to perform a risk analysis. Since users can easily forget that they have set up synchronisations and the synchronisation rules they have may interact in unexpected ways, explicit analysis of these agents remains valuable.

Nevertheless, *in spite of* the difficulty in gaining accurate probabilistic data for the behaviour of humans in the social networks we believe that model-checking does provide a tool which would allow some understanding of the risks of privacy violations and information leaks in social networks. Services which allowed networks to be evaluated on a regular basis in order to asses general risk could be of significant value. While only applied here to very simple examples, we believe the approach described could form the basis for exactly these services.

5.1 Related Work

To the best of our knowledge, this is the first work considering information leakage in the general sense as any information shared on a social network service accessible to persons not directly authorised to access it. Furthermore this is the first attempt to apply formal verification to determine whether, and how often, information leakage occurs.

"Information leakage" is a term typically used in the context of software engineering, to denote the event when a software system designed to be closed for unauthorised parties reveals some information to them nonetheless. In [12] the use of an agent-based approach to facilitate software information leakage is proposed.

Involuntary information leakage within the context of social network services has been considered for sensitive information, such as personal data and location. A study showed that even if people do not directly reveal their personal information in a social networking service, this may happen indirectly with personal information becoming either directly accessible or inferable from accessible information [11]. Multi-agent system (MAS) technology use is proposed in [1] to assess the vulnerability of particular user profiles on a social network service. Specifically, a software agent is associated with each user profile to extract the user's updates and send them to a controller agent which saves the history of each user and analyses it for possible vulnerabilities.

Logic-based representation of social network service users and their interactions is an increasing area of research, although work is mainly aimed at studying the information diffusion in a social network. In particular, [17] proposes a two-dimensional modal logic for reasoning about the changing patterns of knowledge and social relationships in networks. Model-checking as a method for verifying properties of information diffusion in open networks has been studied in [2]. The authors, however, focus on modelling the entire (open dynamic agent) network whereas we are modelling a software agent in a social network service system.

5.2 Further Work

As this paper simply sets out a broad direction, and gives quite simple examples, there is much further work to be done.

We would be interested in extending our system to look at, for instance, how information through different routes (e.g. location information sent to one social network service and information about companions sent to another) can be combined to leak key information in unanticipated ways (*e.g.*, someone can now know the location of your companion). Formal verification would surely be more complex but still viable.

The examples we have provided have been built "by hand" and so it would be advantageous to provide a route whereby (some at least) social networks could be automatically extracted into our formalism.

Finally, we here use a relatively standard model-checker, namely PRISM, as we are not primarily concerned with anything more than the beliefs of our agents. As we move to more complex systems it would be ideal to verify complex BDI behaviours. We have an agent model-checker that is capable of this [6], and indeed this can also be configured to export models to PRISM [5] if probabilistic results are desired. However, it would be ideal to enhance the agent model-checker with explicit content/context constructs in order to facilitate a more direct relationship between our formalism and the model analysed by the tool than we could achieve via a direct translation into PRISM. This would also allow for the practical verification of higher-level properties.

Acknowledgments This work was partially funded through EPSRC Grants EP/L024845 ("Verifiable Autonomy") and EP/N007565 ("Science of Sensor System Software"). The authors would also like to thank Dagstuhl for their facilities and hospitality, something that provided the impetus for this work.

Access to Data The PRISM models used in this work will be made available in the University of Liverpool's Data Catalogue prior to publication and a DOI will be included in the camera ready copy of this paper.

Bibliography

[1] R. Abdulrahman, S. Alim, D. Neagu, D. R. W. Holton, and M. Ridley. Multi Agent System Approach for Vulnerability Analysis of Online Social Network Profiles over Time. *International Journal of Knowledge and Web Intelligence*, 3(3):256–286, December 2012.

[2] F. Belardinelli and D. Grossi. On the Formal Verification of Diffusion Phenomena in Open Dynamic Agent Networks. In *Proc. International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 237–245, 2015.

[3] E. M. Bryant and J. Marmo. The Rules of Facebook Friendship: A two-stage examination of interaction rules in close, casual, and acquaintance friendships. *Journal of Social and Personal Relationships*, 29(8):1013–1035, 2012.

[4] E Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.

[5] L. A. Dennis, M. Fisher, and M. Webster. Two-stage agent program verification. *Journal of Logic and Computation*, 2016.

[6] L. A. Dennis, M. Fisher, M. Webster, and R. H. Bordini. Model Checking Agent Programming Languages. *Automated Software Engineering*, 19(1):5–63, 2012.

[7] M. Fisher, L. Dennis, and A. Hepple. Modular Multi-Agent Design. Technical Report ULCS-09-002, Department of Computer Science, University of Liverpool, 2009.

[8] M. Fisher and T. Kakoudakis. Flexible Agent Grouping In Executable Temporal Logic. In *Proc. 12th Int. Symposium on Languages for Intensional Programming (ISLIP)*. World Scientific Press, 1999.

[9] A. Hepple, L. Dennis, and M. Fisher. A Common Basis for Agent Organisation in BDI Languages. In *Languages, Methodologies and Development Tools for Multi-Agent Systems*, volume 4908 of *LNAI*, pages 71–88. Springer-Verlag, 2008.

[10] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of Probabilistic Real-time Systems. In *Proc. 23rd Int. Conf. Computer Aided Verification (CAV)*, volume 6806 of *LNCS*, pages 585–591. Springer, 2011.

[11] I.-F. Lam, K.-T. Chen, and L.-J. Chen. Involuntary Information Leakage in Social Network Services. In *Advances in Information and Computer Security: Third International Workshop on Security, IWSEC 2008, Kagawa, Japan, November 25-27, 2008. Proceedings*, pages 167–183, Berlin, Heidelberg, 2008. Springer.

[12] Y. C. Lee, S. Bishop, H. Okhravi, and S. Rahimi. Information Leakage Detection in Distributed Systems using Software Agents. In *Proc. IEEE Symposium on Intelligent Agents*, pages 128–135, 2009.

[13] X. Lu, Z. Yu, B. Guo, and X. Zhou. Predicting the Content Dissemination Trends by Repost Behavior Modeling in Mobile Social Networks. *Journal of Network and Computer Applications*, 42:197–207, 2014.

[14] PRISM: Probabilistic Symbolic Model Checker. `http://www.prismmodelchecker.org`. Accessed 2013-05-31.

[15] A. S. Rao and M. P. Georgeff. Modelling Agents within a BDI-Architecture. In *Proc. Int. Conf. Principles of Knowledge Representation and Reasoning (KR)*. Morgan Kaufmann, 1991.

[16] A. S. Rao and M. P. Georgeff. BDI Agents: from Theory to Practice. In *Proc. 1st Int. Conf. Multi-Agent Systems (ICMAS)*, pages 312–319, 1995.

[17] J. Seligman, F. Liu, and P. Girard. Facebook and the Epistemic Logic of Friendship. In *Proc. 14th Conf. Theoretical Aspects of Rationality and Knowledge (TARK)*, 2013.

[18] M. Slavkovik, L. Dennis, and M. Fisher. An Abstract Formal Basis for Digital Crowds. *Distributed and Parallel Databases*, 33(1):3–31, 2015.

[19] C. Stirling. Modal and Temporal Logics. In *Handbook of Logic in Computer Science*. Oxford University Press, 1992.

[20] M. Wooldridge and N. R. Jennings. Intelligent Agents: Theory and Practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.

Monitoring Opportunism in Multi-Agent Systems *

Jieting Luo[1], John-Jules Meyer[2], and Max Knobbout[3]

[1,2]Utrecht University, Utrecht, the Netherlands
[3] Delft University of Technology, the Netherlands
{J.Luo,J.J.C.Meyer}@uu.nl,
M.Knobbout@tudelft.nl

Abstract. Opportunism is a behavior that causes norm violation and promotes own value. In the context of multi-agent systems, we constrain such a selfish behavior through setting enforcement norms. Because opportunistic behavior cannot be observed indirectly, there has to be a monitoring mechanism that can detect the performance of opportunistic behavior in the system. This paper provides a logical framework based on the specification of actions to specify monitoring aproaches for opportunism. We investigate how to evaluate agents' actions to be opportunistic with respect to different forms of norms when those actions cannot be observed directly, and study how to reduce the monitoring cost for opportunism.

1 Introduction

Consider a common social scenario. A seller sells a cup to a buyer and it is known by the seller beforehand that the cup is actually broken. The buyer buys the cup without knowing it is broken. Since the buyer's value gets demoted, the behavior performed by the seller is usually forbidden by the social norm. Such a social behavior intentionally performed by the seller is first named opportunistic behavior (or opportunism) by economist Williamson [10]. It is a typical social behavior that is motivated by self-interest and takes advantage of knowledge asymmetry about the behavior to achieve own gains, regardless of the principles [6]. This definition implies that, given a social context, opportunistic behavior results in promoting own value while demoting social value. Therefore, it is prohibited by norms in most societies. In the context of multiagent systems, we constrain such a selfish behavior through setting enforcement norms, in the sense that agents receive a corresponding sanction when they violate the norm. On the one hand, it is important to detect it, as it has undesirable results for the participating agents. On the other hand, as opportunism is always in the form of cheating, deception and betrayal, meaning that the system does not know what the agent performs or even the motivation behind it (for example, in a distributed system), opportunistic behavior cannot be observed indirectly. Therefore, there has to be a monitoring mechanism that can detect the performance of opportunistic behavior in the system.

This paper provides a logical framework based on the specification of actions to monitor opportunism. In particular, we investigate how to evaluate agents' actions to

* The short paper version of this paper was accepted for ECAI 2016, The Hague.

be opportunistic with respect to different forms of norms when those actions cannot be observed directly, and explore how to reduce the monitoring cost for opportunism. We study formal properties of our monitoring approaches in order to determine whether it is effective in the sense that whenever an action is detected to be opportunistic, it was indeed opportunistic, and that whenever an action was opportunistic, it is indeed detected.

2 Framework

Since monitors cannot observe the performance of opportunism directly, the action can only be identified through the information about the context where the action can be performed and the property change in the system, which is called *action specification* [8] or *action description* [4]. Usually an action can be specified through its precondition and its effect (postcondition): the precondition specifies the scenario where the action can be performed whereas the postcondition specifies the scenario resulting from performing the action. For example, the action, dropping a glass to the ground, can be specified as holding a glass as its precondition and the glass getting broken as its effect. Therefore, we assume that every action has a pair of the form $\langle \psi_p^a, \psi_e^a \rangle$, where ψ_p^a is the precondition of action a and ψ_e^a is the effect of performing action a in the context of ψ_p^a, both of which are propositional formulas. Sometimes a particular action a can have different results depending on the context in which it is performed. Based on this idea, we argue that action a can be represented through a set of pairs $D(a) = \{\langle \psi_p^a, \psi_e^a \rangle, ...\}$, each element indicating its precondition and its corresponding effect. The absence of a preconditioon means that the performance of the action is not context-dependent.

In this paper, the models that we use are transition systems, which consist of agents Agt, states S, actions Act and transitions $\mathcal{R}$ between states by actions. When an action $a \in Act$ is performed in a certain state s, the system might progress to a different state s' in which different propositions might hold. We also extend the standard framework with an observable accessibility relation $\mathcal{M}$. Note that in this paper we don't talk about synchronous actions for simplifying our model, meaning that we assume there is only one action to execute in every state. Moreover, actions are deterministic; the same action performed in the same state will always result in the same new state. Formally,

Definition 2.1. *Let $\Phi = \{p, q, ...\}$ be a finite set of atomic propositional variables. A monitoring transition system over Φ is a tuple $\mathcal{I} = (Agt, S, Act, \pi, \mathcal{M}, \mathcal{R}, s_0)$ where*

- *Agt is a finite set of agents;*
- *S is a finite set of states;*
- *Act is a finite set of actions;*
- *$\pi : S \to \mathcal{P}(\Phi)$ is a valuation function mapping a state to a set of propositions that are considered to hold in that state;*
- *$\mathcal{M} \subseteq S \times S$ is a reflexive, transitive and symmetric binary relation between states, that is, for all $s \in S$ we have $s\mathcal{M}s$; for all $s, t, u \in S$ $s\mathcal{M}t$ and $t\mathcal{M}u$ imply that $s\mathcal{M}u$; and for all $s, t \in S$ $s\mathcal{M}t$ implies $t\mathcal{M}s$; $s\mathcal{M}s'$ is interpreted as state s' is observably accessible from state s;*

- $\mathcal{R} \subseteq S \times Act \times S$ *is a relation between states with actions, which we refer to as the transition relation labelled with an action; since we have already introduced the notion of action specification, a state transition* $(s, a, s') \in \mathcal{R}$ *if there exists a pair* $\langle \psi_p^a, \psi_e^a \rangle \in D(a)$ *such that* ψ_p^a *is satisfied in state* s *and* ψ_e^a *is satisfied in state* s'*, and both* ψ_p^a *and* ψ_e^a *are evaluated in the conventional way of classical propositional logic; since actions are deterministic, sometimes we also denote state* s' *as* $s\langle a \rangle$ *for which it holds that* $(s, a, s\langle a \rangle) \in \mathcal{R}$*;*
- $s_0 \in S$ *denotes the initial state.*

Norms are regarded as a set of constraints on agents' behavior. More precisely, a norm defines whether a possible state transition by an action is considered to be illegal or not. The same as [1], we simply consider a norm as a subset of $\mathcal{R}$ that is decided by the designers of the system. Formally,

Definition 2.2 (Norm). *A norm* η *is defined as a subset of* $\mathcal{R}$*, i.e.* $\eta \subseteq \mathcal{R}$*. Intuitively, given a state transition* (s, a, s')*,* $(s, a, s') \in \eta$ *means that transition* (s, a, s') *is forbidden by norm* η*. We say* (s, a, s') *is an* η*-violation if and only if* $(s, a, s') \in \eta$*. Otherwise,* (s, a, s') *is an* η*-compliant.*

From the way that we define a norm, we can realize two extreme cases: if norm η is an empty set, all the possible state transitions are η-compliant; and it is also possible that a norm leads to states with no legal successor, which means that agents can only violate the norm.

The logical language we use in this paper is propositional logic $\mathcal{L}_{prop}$ extended with action modality, denoted as $\mathcal{L}_{modal}$. The syntax of $\mathcal{L}_{modal}$ is defined by the following grammar:

$$\varphi ::= p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \langle a \rangle \varphi$$

where $p \in \Phi$ and $a \in Act$. The semantics of $\mathcal{L}_{modal}$ are given with respect to the satisfaction relation "$\models$". Given a monitoring transition system $\mathcal{I}$ and a state s in $\mathcal{I}$, a formula φ of the language can be evaluated in the following way:

- $\mathcal{I}, s \models p$ iff $p \in \pi(s)$;
- $\mathcal{I}, s \models \neg\varphi$ iff $\mathcal{I}, s \not\models \varphi$;
- $\mathcal{I}, s \models \varphi_1 \vee \varphi_2$ iff $\mathcal{I}, s \models \varphi_1$ or $\mathcal{I}, s \models \varphi_2$;
- $\mathcal{I}, s \models \langle a \rangle \varphi$ iff $\exists s'$ such that $(s, a, s') \in \mathcal{R}$ and $\mathcal{I}, s' \models \varphi$;

Other classical logic connectives (e.g.,"$\wedge$", "$\rightarrow$") are assumed to be defined as abbreviations by using $\neg$ and $\vee$ in the conventional manner. We write $\mathcal{I} \models \varphi$ if $\mathcal{I}, s \models \varphi$ for all $s \in S$, and $\models \varphi$ if $\mathcal{I} \models \varphi$ for all monitoring transition systems $\mathcal{I}$.

Given the language $\mathcal{L}_{modal}$, a norm η can be defined in a more specific way such that it contains all the state transitions that are forbidden by norm η. Norms are described in various ways so that they can represent the forbidden behaviors explicitly. Below we define three forms of norms: $\eta(\varphi, \psi)$, $\eta(\varphi, a)$ and $\eta(\varphi, a, \psi)$, each following an example for better understanding. Of course, it is only a choice in this paper and more forms of norms can be described and constructed based on our logical framework.

- **Norm** $\eta(\varphi, \psi)$ Let φ and ψ be two propositional formulas and $\mathcal{I}$ be a monitoring transition system. A norm $\eta(\varphi, \psi)$ is defined as the set $\eta_{\mathcal{I}}(\varphi, \psi) = \{(s, a, s') \in$

$\mathcal{R} \mid \mathcal{I}, s \vDash \varphi \wedge \langle a \rangle \psi\}$ In the rest of the paper, we will write $\eta(\varphi, \psi)$ for short. This is the most simple form. The interpreted meaning of a norm $\eta(\varphi, \psi)$ is simply that it is forbidden to achieve ψ in the states satisfying φ (φ-state) by any actions. The forbidden actions are implicitly indicated in this type of norms. For example, it is forbidden to keep the light on when everybody is sleeping, no matter you turn on the flashlight or the lamp or lighten the candle.

- **Norm** $\eta(\varphi, a)$ Let φ be a propositional formula, a be an action, and $\mathcal{I}$ be a monitoring transition system. A norm (φ, a) is defined as the set $\eta_{\mathcal{I}}(\varphi, a) = \{(s, a', s') \in \mathcal{R} \mid \mathcal{I}, s \vDash \varphi \; and \; a' = a\}$. In the rest of the paper, we will write $\eta(\varphi, a)$ for short. The interpreted meaning of a norm $\eta(\varphi, a)$ is that it is forbidden to perform action a in a φ-state. This is the most common form in which the action and the context where the action is forbidden are explicitly represented, regardless of the effect that the action brings about. For example, it is forbidden to smoke in a non-smoking area.
- **Norm** $\eta(\varphi, a, \psi)$ Let φ and ψ be two propositional formulas, a be an action, and $\mathcal{I}$ be a monitoring transition system. A norm (φ, a, ψ) is defined as the set $\eta_{\mathcal{I}}(\varphi, a, \psi) = \{(s, a', s') \in \mathcal{R} \mid \mathcal{I}, s \vDash \varphi \wedge \langle a' \rangle \psi \; and \; a' = a\}$. In the rest of the paper, we will write $\eta(\varphi, a, \psi)$ for short. The interpreted meaning of a norm $\eta(\varphi, a, \psi)$ is that it is forbidden to perform action a in φ-state to achieve ψ. In this type of norms, the action, the context that the action is forbidden and the effect that the action will bring about are all represented explicitly. For example, in China it is forbidden to buy a house based on mortgage when you already own one.

Sometimes, propositional formula φ, which is indicated in three types of norms above, is called the precondition of an action [4]. However, it should be distinguished from the precondition ψ_p we introduced in action pairs. φ is used to characterize the context where the action(s) is forbidden to perform by the system, whereas ψ_p is used to represent in which situation the action can be physically performed. Certainly there are relationships between φ and ψ_p, which will be investigated in our monitoring mechanism for opportunism.

3 Defining Opportunism

Before we propose our monitoring mechanism for opportunism, we should formally define opportunism from the perspective of the system so that the system knows what to detect for monitoring opportunism. In our previous paper [6] we emphasizes opportunistic behavior is performed by intent rather than by accident. However, monitors cannot read agents' mental states, so for monitoring we assume that agents violate the norms by intention from a pragmatic perspective. For example, we always assume that speeding is performed with intention. In this paper we remove all the references to the mental states from the formal definition of opportunism in our previous paper [6], assuming that the system can tell agents' value promotion/demotion causing by an action. In a sentence, from the perspective of the system, opportunistic behavior performed by an agent in a social context can be simply defined as a behavior that causes norm violations and promotes his own value.

Opportunistic behavior results in promoting agents' own value, which can be interpreted as that opportunistic agents prefer the state that results from opportunistic behavior rather than the initial state. For having preferences over different states, we argue that agents always evaluate the truth value of specific propositions in those states based on their value systems. For instance, the seller tries to see whether he gets the money from selling a broken cup in order to have a preference on the states before and after the transaction. After the transaction, the seller's value gets promoted, because the proposition he verifies (whether he gets the money) based on his value system becomes true. Based on this interpretation, we first define a function $EvalRef$:

Definition 3.1 (Evaluation Reference). *Let V be a set of agents' value systems, S be a finite set of states, and Φ be a finite set of atomic propositions, $EvalRef : V \times S \times S \rightarrow \Phi$ is a function named Evaluation Reference that returns a proposition an agent refers to for specifying his preference over two states.*

This function means that the proposition is dependent on the value system and the two states. For simplicity, we assume that for value promotion the truth value of the proposition that agents refer to changes from false to true in the state transition. For example, assuming that proposition p represents the seller earns money, the seller promotes his value in the way of bringing about p through selling a broken cup. Based on this assumption, we can define *Value Promotion*, which is another important element of opportunistic behavior.

Definition 3.2 (Value Promotion). *Given two states s and s', and an agent's value system V, his value gets promoted from state s to s', denoted as $s <_V s'$, iff $s \vDash \neg p$ and $s' \vDash p$, where $p = EvalRef(V, s, s')$.*

As we already introduced the notion of value for defining opportunism, it is natural to extend our logical setting with value systems. We define a tuple of the form $V = (V_1, V_2, ..., V_{|Agt|})$ as agents' value systems. Now the syntax of $\mathcal{L}_{modal}$ still follows the one we defined above, and the semantics with respect to the satisfaction relation become of the form $\mathcal{I}, V, s \vDash \varphi$ but is still defined in the same way as above.

Now we are ready to formalize opportunism from the perspective of the system. Note that, comparing to the definition of opportunism in our previous work, we remove all the references to mental states (knowledge, intention) because it is impossible for monitors to detect any mental states, but we assume that the system can reason about agents' value promotion/demotion by an action based on the corresponding value systems. Firstly, we extend our language to also include $Opportunism(\eta, a)$, and then we extend the satisfaction relation such that the following definition holds.

Definition 3.3 (Opportunism). *Given a monitoring transition system $\mathcal{I}$ with a value system set V and a norm η, an action a performed by agent i in state s being opportunistic behavior is defined as follows: $\mathcal{I}, V, s \vDash Opportunism(\eta, a)$ iff state transition $(s, a, s\langle a\rangle) \in \eta$ and $s <_{V_i} s\langle a\rangle$.*

Intuitively, opportunism is a state transition which is an η-violation. Besides, the state transition also promotes the value of the agent who performs action a (agent i) by bringing about p, which is the proposition that the agent refers to for having preference

over state s and $s\langle a\rangle$. Action a performed in state s, more essentially state transition $(s, a, s\langle a\rangle)$, is opportunistic behavior from the perspective of the system. We illustrate this definition through the following example.

Example 1 (Selling a Broken Cup). Consider the example of selling a broken cup in Figure 1. A seller sells a cup to a buyer. It is known only by the seller beforehand that the cup is actually broken. The buyer buys the cup, but of course gets disappointed when he uses it. Here the state transition is denoted as $(s, sell(brokencup), s')$. Given a social norm $\eta(\top, sell(brokencup))$ interpreted as it is forbidden to sell broken cups in any circumstance, the seller's behavior violates norm η. Moreover, based on the value system of the seller, his value gets promoted after he earns money from the transition $(EvalRef(V_s, s, s') = hasmoney(seller),\ \mathcal{I}, V, s \vDash \neg hasmoney(seller),\ \mathcal{I}, V, s' \vDash hasmoney(seller))$. Therefore, the seller performed opportunistic behavior to the buyer from the perspective of the system.

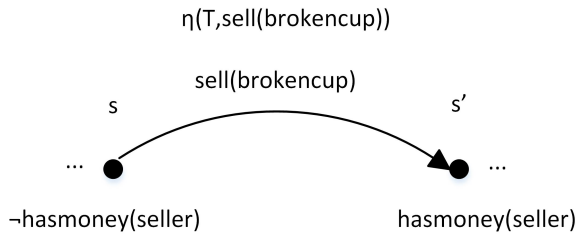


Fig. 1. Opportunistic behavior of selling a broken cup

4 Monitoring Opportunism

We propose a monitoring mechanism for opportunism in this section. A monitor is considered as an external observer that can evaluate a state transition with respect to a given norm. However, a monitor can only verify state properties instead of observing the performance of actions directly. Our approach to solve this problem is to check how things change in a given state transition and reason about the action taking place in between. Here we assume that our monitors are always correct, which means that the verification for state properties can always be done perfectly. In general, we consider monitoring as a matter of observing the physical world with an operator m such that $m(\varphi)$ is read as "φ is detected" for an arbitrary property φ.

We first define a state monitor m_{state}, which can evaluate the validity of a given property in a given state. Because a monitor can be seen as an external observer that can observe agents' activities according to the model, we define state monitors in this paper in a similar way as we define knowledge in epistemic logic, and correspondingly adopt S5 properties. We extend the language to also include $m_{state}(\varphi)$ and the satisfaction relation such that the following definition holds.

Definition 4.1 (State Monitors). *Given a propositional formula φ, a set of value systems V and a monitoring transition system $\mathcal{I}$, a state monitor m_{state} for φ over $\mathcal{I}$ is defined as follows: $\mathcal{I}, V, s \vDash m_{state}(\varphi)$ iff for all s' $s\mathcal{M}s'$ implies $\mathcal{I}, V, s' \vDash \varphi$. Sometimes we will write $m_{state}(\varphi)$ for short if clear from the context.*

Note that we define state monitors with the form $\mathcal{I}, V, s \vDash \phi$ for being consistent with the definitions in the rest of the paper, even though it is not relevant to value. Because $\mathcal{M}$-relation is reflexive, we have the validity $\vDash m_{state}(\varphi) \rightarrow \varphi$, meaning that what the state monitor detects is always considered to be true.

State monitors are the basic units in our monitoring mechanism. We can combine state monitors to check how things change in a given state transition and evaluate it with respect to a given set of norms. In Section 2, we introduced three forms of norms through which certain agents' behaviors are forbidden by the system. As we defined in Section 3, opportunistic behavior performed by an agent is a behavior that causes norm violations and promotes his own value, that is, opportunism is monitored with respect to a norm and a value system of an agent. Based on this definition, we design different monitoring opportunism approaches with respect to different forms of norms and discuss in which condition opportunism can be perfectly monitored. It is worth stressing that one important issue of this paper is to have an effective monitoring mechanism for opportunism in the sense that whenever an action is detected to be opportunistic, it was indeed opportunistic, and that whenever an action was opportunistic, it is indeed detected. Therefore, we will discuss this issue every time we propose a monitoring approach. We extend the language to also include $m_{opp}(\eta, a')$ and the satisfaction relation such that the following definition holds.

Definition 4.2 (Monitoring Opportunism with Norm $\eta(\varphi, \psi)$). *Given a monitoring transition system $\mathcal{I}$, a value system set V, a norm $\eta(\varphi, \psi)$ and an action a' performed by agent i in state s, whether action a' is opportunistic behavior can be monitored through a combination of state monitors as follows:*

$$\mathcal{I}, V, s \vDash m_{opp}((\varphi, \psi), a') := m_{state}(\varphi) \wedge \langle a' \rangle m_{state}(\psi)$$

where

$$\mathcal{I} \vDash \varphi \rightarrow \neg p,\ \mathcal{I} \vDash \psi \rightarrow p,\ \textit{and}\ p = \mathit{EvalRef}(V_i, s, s\langle a' \rangle)$$

In order to detect whether action a' is opportunistic behavior in state s, we check if the state transition $(s, a', s\langle a' \rangle)$ is forbidden by norm $\eta(\varphi, \psi)$: because the interpreted meaning of norm $\eta(\varphi, \psi)$ is that it is forbidden to achieve ψ in φ-state by any actions, we check whether propositional formulas φ and ψ are successively satisfied in a state transition. Moreover, we assume the following implications in our model that φ implies $\neg p$ and ψ implies p, where proposition p is the proposition that agent i who performs action a' looks at based on his value system V_i. Since state s and $s\langle a' \rangle$ are not given and our monitors can only have partial information about the two states, we have a candidate set of states for state s and a candidate set of states for state $s\langle a' \rangle$ and any two states from them satisfy the resulting property of function $\mathit{EvalRef}$, which means that given the partial information the execution of action a' in state s brings about p thus promoting agent i's value. The forbidden actions are not explicitly stated in the norm. Therefore,

although the monitors cannot observe the performance of opportunistic behavior, it still can be perfectly detected with respect to norm $\eta(\varphi, \psi)$, which can be expressed by the following proposition:

Proposition 4.1. *Given a transition system $\mathcal{I}$, a norm $\eta(\varphi, \psi)$, and an action a' performed by agent i in state s, action a' is detected to be opportunistic with respect to $\eta(\varphi, \psi)$ in state s over $\mathcal{I}$ if and only if action a' was indeed opportunistic:*

$$\mathcal{I}, V, s \vDash Opportunism((\varphi, \psi), a') \leftrightarrow m_{opp}((\varphi, \psi), a')$$

Proof. It trivially holds because the monitors detect exactly what the norm indicates and they are assumed to be correct.

Definition 4.3 (Monitoring Opportunism with Norm $\eta(\varphi, a)$). *Given a monitoring transition system $\mathcal{I}$, a value system set V, a norm $\eta(\varphi, a)$, and a pair $\langle \psi_p^a, \psi_e^a \rangle$ of action a ($\langle \psi_p^a, \psi_e^a \rangle \in D(a)$ and $\varphi \wedge \psi_p^a$ is satisfiable on I), whether action a' performed by agent i in state s is opportunistic behavior can be monitored through a combination of state monitors as follows:*

$$\mathcal{I}, V, s \vDash m_{opp}((\varphi, a), \langle \psi_p^a, \psi_e^a \rangle, a') := m_{state}(\varphi \wedge \psi_p^a) \wedge \langle a' \rangle m_{state}(\psi_e^a)$$

where

$$\mathcal{I} \vDash \varphi \wedge \psi_p^a \rightarrow \neg p,\ \mathcal{I} \vDash \psi_e^a \rightarrow p,\ \textit{and}\ p = EvalRef(V_i, s, s\langle a' \rangle)$$

In order to check whether action a' is opportunistic behavior (violates norm $\eta(\varphi, a)$ and promotes own value), we verify if action a' is performed in a φ-state. Besides, we check if action a' is the action that the norm explicitly states. Since the monitors cannot observe the performance of action a', we only can identify action a' to be possibly action a by checking if formulas ψ_p^a and ψ_e^a are successively satisfied in the state transition by action a', where ψ_p^a is action a's precondition and ψ_e^a is the corresponding effect. Similar to norm $\eta(\varphi, \psi)$, we assume that $\varphi \wedge \psi_p^a$ implies $\neg p$ and ψ_e^a implies p, where p is the proposition that agent i refers to based on his value system V_i. Again, with this approach we have a candidate set of states for state s and a candidate set of states for state $s\langle a' \rangle$ and any two states from them satisfy the resulting property of function $EvalRef$, which means that given the partial information the execution of action a' in state s brings about p thus promoting agent i's value.

Given a norm and an agent's value system, we can evaluate whether a state transition by an action is opportunistic behavior. However, since the monitors can only verify state properties instead of observing the performance of the action directly, we cannot guarantee that an action that is detected to be opportunistic was indeed opportunistic, which is given by the following proposition:

Proposition 4.2. *Given a monitoring transition system $\mathcal{I}$, a value system set V, a norm $\eta(\varphi, a)$, a pair $\langle \psi_p^a, \psi_e^a \rangle$ of action a ($\langle \psi_p^a, \psi_e^a \rangle \in D(a)$ and $\varphi \wedge \psi_p^a$ is satisfiable on I), an action a' performed by agent i in state s, action a' that is detected to be opportunistic was possibly opportunistic, which is characterized as*

$$\mathcal{I}, V, s \nvDash m_{opp}((\varphi, a), \langle \psi_p^a, \psi_e^a \rangle, a') \rightarrow Opportunism((\varphi, a), a')$$

Proof. This is because pair $\langle \psi_p^a, \psi_e^a \rangle$ might not be unique for action a within the actions that can be performed in φ-state. That is, we have a set of actions $Act' = \{a' \in Act \mid \mathcal{I}, V, s \vDash m_{state}(\varphi \wedge \psi_p^a) \wedge \langle a' \rangle m_{state}(\psi_e^a)\}$, and action a indicated in norm η is one of them ($a \in Act'$).

Given this problem, we want to investigate in which case or with what requirement the action that is detected by the opportunism monitor is indeed opportunistic behavior. We first introduce a notion of action adequacy. An action $a \in Act$ is called adequate to achieve ψ at state $s \in S$ if and only if there exists a pair of $\langle \psi_p^a, \psi_e^a \rangle$ in $D(a)$ such that $\mathcal{I}, V, s \vDash \psi_p^a$ and $\mathcal{I}, V, s \vDash \langle a \rangle(\psi_e^a \rightarrow \psi)$ hold. $Ad(s, \psi)$ is a function that maps each state ($s \in S$) and a propositional formula ψ to a non-empty subset of actions, denoting the actions that are adequate to achieve ψ in state s, thus we have $Ad(s, \psi) \in \mathcal{P}(Act)$. And then we have the following proposition:

Proposition 4.3. *Given a monitoring transition system $\mathcal{I}$, a value system set V, a norm $\eta(\varphi, a)$, a pair $\langle \psi_p^a, \psi_e^a \rangle$ of action a ($\langle \psi_p^a, \psi_e^a \rangle \in D(a)$ and $\varphi \wedge \psi_p^a$ is satisfiable on I), an action a' performed by agent i in state s, the following statements are equivalent:*

1. $\mathcal{I}, V, s \vDash m_{opp}((\varphi, a), \langle \psi_p^a, \psi_e^a \rangle, a') \leftrightarrow Opportunism((\varphi, a), a')$;
2. *there exists only one action* $a \in \bigcup_{s \in S'} Ad(s, \top)$ *that has pair* $\langle \psi_p^a, \psi_e^a \rangle$, *where* $S' = \{s \in S \mid \mathcal{I}, V, s \vDash \varphi\}$.

Proof. From 1 to 2: Statement 1 implies that action a' that is detected to be opportunistic was indeed opportunistic. If it holds, then $a' = a$. Because we identify action a with pair $\langle \psi_p^a, \psi_e^a \rangle$, $a' = a$ implies that pair $\langle \psi_p^a, \psi_e^a \rangle$ is unique for action a within the set of actions $\bigcup_{s \in S'} Ad(s, \top)$. In other words, we cannot find one more action in $\bigcup_{s \in S'} Ad(s, \top)$ that also has a pair $\langle \psi_p^a, \psi_e^a \rangle$. From 2 to 1: If action pair $\langle \psi_p^a, \psi_e^a \rangle$ is unique for action a within $\bigcup_{s \in S'} Ad(s, \top)$, then once the pair is detected in the state transition we can deduce that $a' = a$. Hence, action a' is indeed opportunistic behavior. And from the proof of proposition 4.2 we can see that action a is within the set of actions that are detected to be opportunistic, so if action a' was opportunistic behavior then it is indeed detected.

We can also derive a practical implication from this proposition: in order to better monitor opportunistic behavior, we should appropriately find an action pair $\langle \psi_p^a, \psi_e^a \rangle$ such that the possible actions in between can be strongly restricted and minimized. Assume that we use monitor $m_{opp}((\varphi, a), \langle \top, \top \rangle, a')$, the possibility that the opportunism monitor makes an error is extremely high, because every action that is available in φ-state will be detected to be opportunistic behavior.

Definition 4.4 (Monitoring Opportunism with Norm $\eta(\varphi, a, \psi)$). *Given a monitoring transition system $\mathcal{I}$, a value system set V, a norm $\eta(\varphi, a, \psi)$, and a pair $\langle \psi_p^a, \psi_e^a \rangle$ of action a ($\langle \psi_p^a, \psi_e^a \rangle \in D(a)$ and $\varphi \wedge \psi_p^a$ and $\psi \wedge \psi_e^a$ are satisfiable on $\mathcal{I}$), whether action a' performed by agent i in state s is opportunistic behavior can be monitored through a combination of state monitors as follows:*

$$\mathcal{I}, V, s \vDash m_{opp}((\varphi, a, \psi), \langle \psi_p^a, \psi_e^a \rangle, a') := m_{state}(\varphi) \wedge \langle a' \rangle m_{state}(\psi) \wedge m_{state}(\psi_p^a) \wedge \langle a' \rangle m_{state}(\psi_e^a)$$

where

$$\mathcal{I} \vDash \varphi \wedge \psi_p^a \rightarrow \neg p,\ \mathcal{I} \vDash \psi \wedge \psi_e^a \rightarrow p,\ \textit{and}\ p = \mathit{EvalRef}(V_i, s, s\langle a' \rangle)$$

In order to check whether action a' is opportunistic behavior (violates norm $\eta(\varphi, a, \psi)$ and promotes own value), we verify if action a' is performed in a φ-state and secondly verify if action a' brings about ψ. Besides, as the forbidden action a is explicitly stated in norm η, we only can identify action a' to be possibly action a by checking if formulas ψ_p^a and ψ_e^a are successively satisfied in the state transition by action a', where ψ_p^a is action a's precondition and ψ_e^a is the corresponding effect. Similar to norm $\eta(\varphi, \psi)$ and $\eta(\varphi, a)$, we assume that $\varphi \wedge \psi_p^a$ implies $\neg p$ and $\psi \wedge \psi_e^a$ implies p, where p is the proposition that agent i refers to based on his value system V_i. Again, with the partial information our monitors have detected we have a candidate set of states for state s and a candidate set of states for state $s\langle a' \rangle$ and any two states from them satisfy the resulting property of function $\mathit{EvalRef}$, which means that given the partial information the execution of action a' in state s brings about p thus promoting agent i's value.

The same as we do with $\eta(\varphi, a)$, we cannot guarantee that an action that is detected to be opportunistic was indeed opportunistic, which is given by the following proposition:

Proposition 4.4. *Given a monitoring transition system $\mathcal{I}$, a value system set V, a norm $\eta(\varphi, a, \psi)$, a pair $\langle \psi_p^a, \psi_e^a \rangle$ of action a ($\langle \psi_p^a, \psi_e^a \rangle \in D(a)$ and $\varphi \wedge \psi_p^a$ and $\psi \wedge \psi_e^a$ are satisfiable on I), action a' that is detected to be opportunistic was possibly opportunistic, which is characterized as*

$$\mathcal{I}, V, s \nvDash m_{opp}((\varphi, a, \psi), \langle \psi_p^a, \psi_e^a \rangle, a') \rightarrow \mathit{Opportunism}((\varphi, a, \psi), a')$$

Proof. Similar to proposition 4.2, it is because pair $\langle \psi_p^a, \psi_e^a \rangle$ might not be unique for action a within the actions that can be performed in φ-state to achieve ψ, and action a indicated in norm η is one of those actions.

Because in our framework the set of state transitions is finite, we can assume that all the possible state transitions are known beforehand. As all the state transitions in our framework are labelled with an action, we introduce a function called $Al(a)$, which maps each action to a non-empty subset of state transitions, denoting all the transitions labelled with action a. Thus we have $Al(a) \in \mathcal{P}(\mathcal{R})$. And then we have the following proposition:

Proposition 4.5. *Given a monitoring transition system $\mathcal{I}$, a value system set V, a norm $\eta(\varphi, a, \psi)$, a pair $\langle \psi_p^a, \psi_e^a \rangle$ of action a ($\langle \psi_p^a, \psi_e^a \rangle \in D(a)$ and $\varphi \wedge \psi_p^a$ and $\psi \wedge \psi_e^a$ are satisfiable on I), and an action a' performed by agent i in state s, the following statements are equivalent:*

1. $\mathcal{I}, V, s \vDash m_{opp}((\varphi, a, \psi), \langle \psi_p^a, \psi_e^a \rangle, a') \leftrightarrow \mathit{Opportunism}((\varphi, a, \psi), a')$;
2. *there exists only one action* $a \in \bigcup_{s \in S'} Ad(s, \psi)$ *that has a pair* $\langle \psi_p^a, \psi_e^a \rangle$, *where* $S' = \{s \in S \mid \mathcal{I}, V, s \vDash \varphi\}$;
3. $\mathcal{R}' = \{(s, a', s') \in \mathcal{R} \mid \mathcal{I}, V, s \vDash \varphi \wedge \psi_p^a \wedge \langle a' \rangle(\psi \wedge \psi_e^a)\} \subseteq Al(a)$.

Proof. The proof for from $1 \Rightarrow 2$ is the same as the proof of proposition 4.3, so we are going to prove from $2 \Rightarrow 3$ and from $3 \Rightarrow 1$. We can consider ψ_p^a and ψ_e^a as two normal propositional formulas. From statement 2 it is clear that $\varphi \wedge \psi_p^a$ and $\psi \wedge \psi_e^a$ are successively satisfied in the state transition. From this we can divide the transitions into two classes: one for the transitions that $\varphi \wedge \psi_p^a$ and $\psi \wedge \psi_e^a$ are successively satisfied (denoted as $\mathcal{R}'$), and the other do not. Since pair $\langle \psi_p^a, \psi_e^a \rangle$ is unique to action a within $\mathcal{R}'$, all the transitions in $\mathcal{R}'$ are labelled with action a. Therefore, $\mathcal{R}'$ is a subset of $Al(a)$. From $2 \Rightarrow 3$ is concluded. From $3 \Rightarrow 1$, if all the transitions in $\mathcal{R}'$ are labelled with action a, then $a' = a$ and we can guarantee that action a' is indeed opportunistic behavior.

Example 1 (continued). We still use the example of selling a broken cup Figure 2 to illustrate our monitoring approach. Here the state transition is denoted as (s, a', s') instead of $(s, sell(brokencup), s')$ because the monitor cannot observe the action directly. Given a social norm $\eta(\top, sell(brokencup))$ and the seller's value system V_s, the system checks whether the seller performed opportunistic behavior. Firstly, the monitor doesn't need to check the context where action a' is performed because action $sell(brokencup)$ is forbidden in any context as norm η says. Secondly, the monitor tries to identify if action a' is indeed $sell(brokencup)$ as norm η indicates: assuming that $\langle hascup(seller) \wedge \neg hasmoney(seller), hascup(buyer) \wedge hasmoney(seller) \rangle$ is the pair we find for action $sell(brokencup)$, we check if both $\mathcal{I}, V, s \vDash m_{state}(hascup(seller))$ and $\mathcal{I}, V, s' \vDash m_{state}(hascup(buyer) \wedge hasmoney(seller))$ hold. Moreover, the information we had for state s and s' implies that the seller's value gets promoted based on the value system V_s, as $EvalRef(V_s, s, s') = hasmoney(seller)$. If they all hold, action a' is detected to be opportunistic behavior. As the action pair we find is unique to action $sell(brokencup)$, action a' is indeed $sell(brokencup)$ thus being opportunistic.

However, if $\langle hascup(seller), hascup(buyer) \rangle$ is the pair that we find for action $sell(brokencup)$, then action a' is not necessarily $sell(brokencup)$ because possibly $a' = give(brokencup)$, meaning that $\langle hascup(seller), hascup(buyer) \rangle$ is not unique to action $sell(brokencup)$.

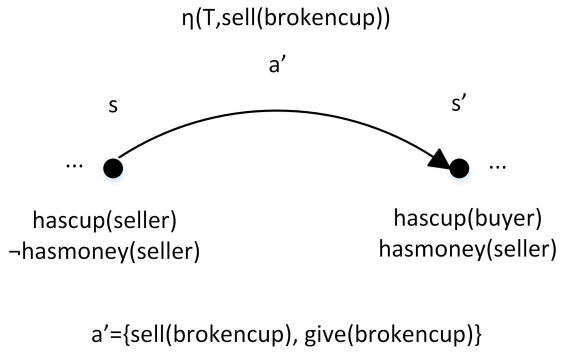


Fig. 2. Monitoring opportunism of selling a broken cup

We proposed three approaches to monitor opportunistic behavior with respect to three different forms of norms. Based on the definitions of three approaches, the following validities hold: given an action a',

$$\mathcal{I}, V \vDash m_{opp}((\varphi, a, \psi), \langle \psi_p^a, \psi_e^a \rangle, a') \rightarrow m_{opp}((\varphi, \psi), a')$$
$$\mathcal{I}, V \vDash m_{opp}((\varphi, a, \psi), \langle \psi_p^a, \psi_e^a \rangle, a') \rightarrow m_{opp}((\varphi, a), \langle \psi_p^a, \psi_e^a \rangle, a')$$

The interpreted meaning of the first validity is that, if action a' is detected to be opportunistic behavior with respect to norm $\eta(\varphi, a, \psi)$, then it will be also detected to be opportunistic behavior with respect to norm $\eta(\varphi, \psi)$. Similar with the second validity. This is simply because, the less information the norm gives, the more actions are forbidden to perform. The state transitions that violate norm $\eta(\varphi, a, \psi)$ is the subset of the state transitions that violate norm $\eta(\varphi, \psi)$ or $\eta(\varphi, a)$. This gives us an implication that the approach to monitor opportunistic behavior with respect to $\eta(\varphi, a, \psi)$ can be used to monitor the other two ones, because $\eta(\varphi, a)$ can be represented as $\eta(\varphi, a, \top)$ and $\eta(\varphi, \psi)$ can be represented as $\eta(\varphi, a, \psi)(\forall a \in Act)$. But there is monitoring cost involved. Apparently the approach with respect to $\eta(\varphi, a, \psi)$ is the most costly one because we need to check more things compared to the other two ones. We will study our monitoring mechanism with cost in the next section.

5 Monitoring Cost for Opportunism

For designing a monitoring mechanism, we not only think about whether it can perfectly detect agents' activities, but also consider if it is possible to decrease the cost involved in the monitoring process. In this section, we will study monitoring cost for opportunism based on the approaches we proposed in the previous section.

There is always cost involved when we monitor something, and the cost depends on what we want to check and how accurate the result we want to get. For example, checking DNA is more expensive than checking a finger print. Our basic idea in this paper is that a monitor is considered as an external observer to verify state properties, and that given a set of propositional formulas X as state properties, we verify the conjunction of formulas from X through combining state monitors. Therefore, we define monitoring cost through a function $c : \mathcal{L}_{prop} \rightarrow \mathbb{R}^+$. Intuitively, given a state property denoted by a propositional formula φ, function $c(\varphi)$ returns a positive real number representing the cost that it takes to verify φ. Such costs can be deduced from expert knowledge and are assumed to be given.

Definition 5.1 (Monitoring Cost). *Cost c over state properties $\mathcal{L}_{prop}$ is a function $c : \mathcal{L}_{prop} \rightarrow \mathbb{R}^+$ that maps a propositional formula to a positive real number. Given a set of propositional formulas X, we also define $c(X) := \sum_{\varphi \in X} c(\varphi)$ for having the cost of monitoring a set X.*

Given a set of propositional formulas X, the cost of monitoring X is the sum of the cost of verifying each element in X. However, if it holds for $\varphi, \varphi' \in X$ that $\varphi \neq \varphi'$, and $\varphi \rightarrow \varphi'$, then monitoring $X \backslash \{\varphi'\}$ is actually the same as monitoring X: when φ is detected to be true, φ' must be true; when φ is detected to be false, the conjunction based

on X is false. But $c(X\backslash\{\varphi'\})$ is less than $c(X)$ if we logically assume that there is no inference cost [1]. This leads us to have the following definition *Largest Non-inferential Subset*:

Definition 5.2 (Largest Non-inferential Subset). *Given a monitoring transition system $\mathcal{I}$ and a set of formulas X, let $X_{\mathcal{I}}$ be the largest non-inferential subset such that for all $\varphi \in X_{\mathcal{I}}$ there is no $\varphi' \in X_{\mathcal{I}}$ with $\varphi \neq \varphi'$ such that $\mathcal{I} \vDash \varphi \rightarrow \varphi'$.*

Proposition 5.1. *Given a monitoring transition system $\mathcal{I}$, a set of formulas X and its largest non-inferential subset $X_{\mathcal{I}}$, it holds that $c(X_{\mathcal{I}}) \leq c(X)$.*

Proof. It holds obviously because $X_{\mathcal{I}}$ is a subset of X.

Therefore, given a set of propositional formulas we want to verify, we always look for its largest non-inferential subset before checking anything in order to reduce the monitoring cost. Certainly, there are more properties among those formulas but we leave them for future study.

For reducing monitoring cost, it is also important to verify a set of propositional formulas $X = \{\varphi_1, ..., \varphi_n\}$ in a certain order instead of checking each formula $\varphi_i (1 \leq i \leq n)$ randomly. Besides, given the truth property of conjunction that a conjunction of propositions returns false if and only if there exists at least one false proposition, we can stop monitoring X once a proposition is detected to be false because it has already made the conjunction false, regardless of the truth value of the rest of the propositions. Therefore, it is sensible to sort the propositions in X in ascending order by cost before checking anything, when the sorting cost is much lower than the monitoring cost. In total, we have $n!$ sequences over X. A sequence over X is denoted as $\lambda(X)$ and the set of all the sequences over X is denoted as $L(X)$. In order to study monitoring cost with monitoring order, we first define the function of monitoring cost for a sequence and an ordered sequence by monitoring cost:

Definition 5.3 (Monitoring Cost for Sequences). *Given a set of propositional formulas $X = \{\varphi_1, ..., \varphi_n\}$ and a sequence $\lambda(X)$, the monitoring cost of checking $\lambda(X)$ is defined as follows:*

$$c(\lambda(X)) := \sum_{i=1}^{n} c(\varphi_i) d_i,$$

where

$$d_i = \begin{cases} 0 & \textit{if } m(\varphi_{i-1}) = \textit{false or } d_{i-1} = 0 \ (i > 1); \\ 1 & \textit{otherwise}. \end{cases}$$

With this function of monitoring cost for a sequence, the monitoring process will stop and no more monitoring cost will have after a false proposition is detected. Given the monitoring cost of each proposition, we can sort the propositions in X in ascending order by monitoring cost.

[1] Assuming that inference cost is lower than monitoring cost is logical, as we only need to compute the inference relation among formulas in the machine while monitoring usually requires setting up costly hardwares (such as cameras).

Definition 5.4 (Cost Ordered Sequence). *Given a set of propositional formulas X, a cost ordered sequence X_c is a sequence over X ordered by the monitoring cost of each element in X such that $X_c \in L(X)$ and for $0 \leq i \leq j$ we have $c(X_c[i]) \leq c(X_c[j])$. In general, such a sequence is not unique because it is possible for two propositions to have the same monitoring cost; in this case we choose one arbitrarily.*

A cost ordered sequence X_c represents the monitoring order over X: we follow the order in X_c to check the elements in X one by one. In general, we can reduce the monitoring cost if we follow the cost ordered sequence, which is represented by the following proposition:

Proposition 5.2. *Given a set of propositional formulas X and a cost ordered sequence X_c over X, if formulas in X are independent of each other, the expected value of the monitoring cost of X_c is the lowest in that of any sequence over X, that is, $E(c(X_c)) \leq E(c(\lambda(X)))$, where $\lambda(X) \in L(X)$.*

Proof. Because before detecting we have no knowledge abou the truth value of the formulas in X, the priori probability that each formula $\varphi \in X$ is true is 1/2. Since there are $|X| = n$ propositions in X and each proposition can be detected to be true or false, there are in total 2^n scenarios about the truth value of the propositions in X, and the monitoring cost for each scenario can be calculated according to Definition 5.3. Let us use $Scen(X)$ to denote the set of all the scenarios about the truth value of the propositions in X, and each scenario from $Scen(X)$ denoted as $\hat{\varphi}$, contains for each proposition $\varphi \in X$ either *true* or *false*. Therefore, the expected value of the monitoring cost of any $\lambda(X)$ is formalized as

$$
\begin{aligned}
E(c(\lambda(X))) &= \frac{1}{2^n} \sum_{\hat{\varphi} \in Scen(X)} \sum_{i=1}^{n} c(\varphi_i) d_i \\
&= \frac{1}{2^n} \left(\sum_{i=1}^{n} c(\lambda(X)[i]) + \sum_{i=1}^{n} 2^{n-n} c(\lambda(X)[i]) + ... + 2^{n-1} c(\lambda(X)[1]) \right),
\end{aligned}
$$

where $\sum_{i=1}^{n} c(\lambda(X)[i])$ represents the monitoring cost for the scenario where all the propositions are detected to be true, and $\sum_{i=1}^{n} 2^{n-n} c(\lambda(X)[i])$ represents the monitoring cost for the scenario where all the propositions are detected to be true except the last one, ..., and $2^{n-1} c(\lambda(X)[1])$ represents the monitoring cost for the scenarios where the first proposition is detected to be false. From this equation we can see that the monitoring cost of the propositions at the front of the sequence strongly influence the value of $E(c(\lambda(X)))$: the lower monitoring cost the propositions at the front have, the less value $E(c(\lambda(X)))$ returns. Thus, the expected value of the monitoring cost of X_c is the lowest in all the sequences over X.

Until here we investigated monitoring cost for any finite set of formulas generally. We can apply the above ideas to monitoring opportunism. Recall that opportunism is monitored with respect to a norm and a value system. Given a norm $\eta(\varphi, a, \psi)$ and

a value system V_i, we evaluate a state transition (s, a', s') by checking whether set $X_1 = \{\varphi, \psi_p^a, p\}$ hold in state s, and whether $X_2 = \{\varphi, \psi_e^a, p\}$ hold in state s', where $\langle \psi_p^a, \psi_e^a \rangle \in D(a)$ and $p = \mathit{EvalRef}(V_i, s, s')$. Note that we cannot combine set X_1 and X_2 into one set because we verify the formulas from the two sets in different states. The inferences reltion among the formulas give rise to the relation between different monitoring approaches.

Proposition 5.3. *Given a monitoring transition system $\mathcal{I}$, a value system set V, a norm $\eta(\varphi, a, \psi)$, a pair $\langle \psi_p^a, \psi_e^a \rangle$ of action a ($\langle \psi_p^a, \psi_e^a \rangle \in D(a)$ and $\varphi \wedge \psi_p^a$ and $\psi \wedge \psi_e^a$ are satisfiable on I), and an action a', if $\mathcal{I}, V \vDash (\varphi \rightarrow \psi_p^a) \wedge (\psi \rightarrow \psi_e^a)$, then*

$$\mathcal{I}, V \vDash m_{opp}((\varphi, \psi), a') \rightarrow m_{opp}((\varphi, a, \psi), \langle \psi_p^a, \psi_e^a \rangle, a');$$

if $\mathcal{I}, V \vDash \psi_e^a \rightarrow \psi$, then

$$\mathcal{I}, V \vDash m_{opp}((\varphi, a), \langle \psi_p^a, \psi_e^a \rangle, a') \rightarrow m_{opp}((\varphi, a, \psi), \langle \psi_p^a, \psi_e^a \rangle, a').$$

Proof. If $\mathcal{I} \vDash (\varphi \rightarrow \psi_p^a) \wedge (\psi \rightarrow \psi_e^a)$ holds, we have the largest non-inferential subset of X_1, $(X_1)_{\mathcal{I}} = \{\varphi\}$, and the largest non-inferential subset of X_2, $(X_2)_{\mathcal{I}} = \{\psi\}$, which means that we only need to verify φ in the initial state and ψ in the final state of any state transition. Thus, if action a' is detected to be opportunistic with norm $\eta(\varphi, \psi)$, it is also the case with norm $\eta(\varphi, a, \psi)$. We can prove the second statement similarly.

This proposition implies that when the above inference holds we can monitor opportunism with the approach $m_{opp}((\varphi, \psi), a')$ (or $m_{opp}((\varphi, a), \langle \psi_p^a, \psi_e^a \rangle, a')$) rather than $m_{opp}((\varphi, a, \psi), \langle \psi_p^a, \psi_e^a \rangle, a')$ for saving monitoring cost.

Together with our general ideas about monitoring cost, we propose the following steps to monitor opportunism: given a monitoring transition system $\mathcal{I}$, a value system set V, a norm $\eta(\varphi, (a), (\psi))$ in any form, a pair $\langle \psi_p^a, \psi_e^a \rangle$ and an action a' performed by agent i in state s, in order to check whether action a' is opportunistic behavior,

1. Check if there is any inference in $\mathcal{I}, V$ among the formulas we need to verify in state s $X_1 = \{\varphi, \psi_p^a, p\}$ and $s\langle a' \rangle$ $X_2 = \{\varphi, \psi_e^a, p\}$, find out the largest non-inferential subsets $(X_1)_{\mathcal{I}}$ and $(X_2)_{\mathcal{I}}$, and choose the corresponding monitoring approach;
2. Arrange all the formulas from $(X_1)_{\mathcal{I}}$ and $(X_2)_{\mathcal{I}}$ in a sequence ordered by monitoring cost $((X_1)_{\mathcal{I}} \cup (X_2)_{\mathcal{I}})_c$;
3. Verify all the formulas from $((X_1)_{\mathcal{I}} \cup (X_2)_{\mathcal{I}})_c$ one by one; when one formula is detected to be false, the monitoring process stops and action a' is detected not to be opportunistic behavior; otherwise, it is detected to be opportunistic behavior.

With the above steps, the monitoring cost for opportunism can be reduced in general when the monitoring is performed for lots of times. For a single time of monitoring, we still cannot guarantee that the monitoring cost is reduced with the above steps, as possibly (only) the last formula in the sequence ordered by cost is detected to be false, for which the monitoring cost is the highest compared to any sequence ordered at random.

6 RELATED WORK

Opportunism is a social and economic concept proposed by economist Williamson [10]. The investigation of opportunism in multi-agent system is still new. [6] proposes a formal definition of opportunism based on situation calculus, which forms a theoretical foundation for any further study related to opportunism. Compared to the definition in [6], we remove all the references to mental states for proposing our monitoring approaches, but still captures norm violation and agents' own-value promotion that the system can recognize and reason about.

The specification of actions is a crucial element in our framework and monitoring mechanism. In general, it consists of the precondition of an action that specifies when the action can be carried out and the effect of an action that specifies the resulting state. A lot of logic formalisms are constructed based on this idea, such as Hoare logic [5] and the situation calculus [7]. In Hoare logic, the execution of a program is described through Hoare triple $\{P\}C\{Q\}$, where C is a program, P is the precondition and Q is the postcondition, which is quite close to our approach of action pair $\langle \psi_p^a, \psi_e^a \rangle$. In the situation calculus, the effect of action is specified through successor state axioms, which consist of positive consequences and negative consequences.

Our work is also related to norm violation monitoring. Norms have been used as a successful approach to regulate and organize agents' behaviors [9]. There are various ways of the specification of norms and norm violations such as [2]. Similar to [1], we only consider a norm as a subset of all possible system behaviors. About norm violation monitoring, [3] proposes a general monitoring mechanism for the situation where agents' behaviors cannot be perfectly monitored. Our work is strongly inspired by them, but we focus on the situation where agents' actions cannot be observed directly but can be reasoned about through checking how things change, assuming state properties can be perfectly verified.

7 CONCLUSION

Opportunism is a behavior that causes norm violation and promotes agents' own value. In order to monitor its performance in the system, we developed a logical framework based on the specification of actions. In particular, we investigated how to evaluate agents' actions to be opportunistic with respect to different forms of norms when those actions cannot be observed directly, and studied how to reduce the monitoring cost for opportunism. We proved formal properties aiming at having an effective and cost-saving monitoring mechanism for opportunism. Future work can be done on value: in our monitoring approaches it is assumed that we can reason whether an action promotes/demotes the value with a value system and how things change by the action, but a value system is still like a black box that we still don't know how the propositions we detect relate to a value system. Moreover, in our framework every state transition is labelled with an action and an agent. We can improve the effectiveness of our monitoring mechanism by attaching *capability* to agents. In this way, given an agent with its capability, the possible actions that were performed by the agent can be eliminated. About reducing monitoring cost, more properties among formulas can be studied together with the relations among the formulas we detect for monitoring opportunism.

References

1. Thomas Agotnes, Wiebe Van Der Hoek, JA Rodriguez-Aguilar, Carles Sierra, and Michael Wooldridge, 'On the logic of normative systems', in *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI'07)*, pp. 1181–1186, (2007).
2. Alan Ross Anderson, 'A reduction of deontic logic to alethic modal logic', *Mind*, **67**(265), 100–103, (1958).
3. Nils Bulling, Mehdi Dastani, and Max Knobbout, 'Monitoring norm violations in multi-agent systems', in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pp. 491–498, (2013).
4. J Fiadeiro and T Maibaum, 'Temporal reasoning over deontic specifications', *Journal of Logic and Computation*, **1**(3), 357–395, (1991).
5. Charles Antony Richard Hoare, 'An axiomatic basis for computer programming', *Communications of the ACM*, **12**(10), 576–580, (1969).
6. Jieting Luo and John-Jules Meyer, 'A formal account of opportunism based on the situation calculus', *AI & SOCIETY*, 1–16, (2016).
7. John McCarthy, 'Situations, actions, and causal laws', Technical report, DTIC Document, (1963).
8. Raymond Reiter, *Knowledge in action: logical foundations for specifying and implementing dynamical systems*, MIT press, 2001.
9. Yoav Shoham and Moshe Tennenholtz, 'On the synthesis of useful social laws for artificial agent societies (preliminary report)', in *AAAI*, pp. 276–281, (1992).
10. Oliver E Williamson, 'Markets and hierarchies: analysis and antitrust implications: a study in the economics of internal organization', (1975).

Sanction recognition: A simulation model of extended normative reasoning.

Martin Neumann and **Ulf Lotzmann**[1]

Abstract. The submission describes aspects of an analysis of the violent breakdown of a criminal group, undertaken as part of the project GLODERS (www.gloders.eu) which provided a set of computational tool for analysis and simulation of extortion racket systems (ERS). Task of the research included studying intra-organizational norms within criminal networks and organizations. The analysis of the breakdown of a criminal group revealed theoretical insights in one of the most important theoretical concepts in the social sciences: social norms. The very fact that a criminal group operates outside the state monopoly of violence generates an ambiguity that hampers the recognition of sanctions.

1 INTRODUCTION

The paper describes part of the project GLODERS aimed at developing ICT models for the comprehension of norms regulating the relations between civil society and organized crime as well as the norms regulating the internal relations within organized crime groups. Here we concentrate on the analysis of the intra-organizational norms within criminal groups. For this purpose a participatory modelling approach had been used [1]. In close collaboration with stakeholders from police forces a case study of the collapse of criminal group had been analyzed. The analysis revealed that normative codes of procedure are constituted by violence [2].

This finding provides insight for sociological theory in general: It has often been claimed that punishment is a central mechanism for the constitution of social order [3-5]. In particular in rational choice theories of norms the notion of sanctions is a central theoretical element [6]. Norms are perceived as a certain degree of behaviour regularity within a group that is ensured by sanctioning norm deviation [7]. Thus it is a theory of norm enforcement. While it is acknowledged the individuals might comply with norms even in the absence of sanctions, ultimately the validity of norms refers to the notion of sanctions or at least the risk of being sanctioned as a norm enforcement mechanism [7-10]. Sanctioning deviant behaviour is not only allowed but even requested. Thus sanctions are a central theoretical concept in a rational choice account of norms. However, the notions of sanctions itself is commonly regarded as unproblematic. Typically it is not further explained what behavioural patterns constitute sanctions. Rather they are treated as a theoretical terminus which is introduced to explain further observations such as norm conformity [11-13].

However, the research on criminal groups revealed that empirically sanctions are an ambiguous concept that requires certain preconditions. Namely, ambiguity needs to be resolved by the existence of a legitimate normative code of procedure. The classical role model is the legal code constituted by a state monopoly of violence. However, exactly this precondition is not met in the case of criminal groups as they operate outside the state monopoly of violence. For this reason the violent constitution of a normative code of procedure remains ambiguous: Recognition of sanctions depends on an error prone process of interpretation. The empirical case shows that sanction recognition is an ambiguous process. In the absence of a state monopoly of violence it is likely to be error prone.

During the GLODERS project this interpretation process had been analyzed by simulation, which currently remains a black-box in normative agent-based simulation models.

2 PRIOR RESEARCH

Normative agent-based simulation models can broadly be characterized by two categories: On the one hand models inspired by evolutionary game theory with a theoretical background in rational choice theory. On the other hand models cognitively richer models with a background in artificial intelligence and cognitive science. Certainly this is only a tendency and not a clear-cut disjunction [14].

A game theoretical setting is characterised by a strategic decision situation in which the benefit of the individual decision is dependent on the decision of other agents. In classical cooperation games agents face a binary decision situation to cooperate or defect. Typically agents would yield higher returns of (somehow measured) utility if both (in two player games) or all (in N-player games) agents would cooperate. However, at least in the short term, defection yields higher returns in terms of individual utility if the other agent(s) do not cooperate. Yet, since agents cannot control the decision of the other agent, they are trapped in a non-optimal equilibrium of defection. Individually the players would lose utility values if the decide to cooperate independent of the decision of the other agents. In such a situation a social force of normative prescriptions to cooperate might push agents away from mutual defection.

Agent-based models have extensively been used in evolutionary game theory. Simulation models have been applied in iterated N-person games to study the evolution of norms [14]. Already in 1986 Axelrod has set the frame with a prototypical model of norms and so-called meta-norm games [15]. In the norms game an agent is has the options to defect or not defect. With a certain probability defection can be observed by other agents. These decide to punish or not to punish the defector. In case of punishment the defector

[1] Institute for information systems, University of Koblenz, email (corresponding author): maneumann@uni-koblenz.de

receives a negative payoff while the punishing agent has to pay a certain cost for the effort of punishing to capture the intuition that also punishing involves some efforts. In simulation experiments agents reproduce differentially dependent on their success. It is investigated whether the system of agents in this game arrive at a state of in which no defecting agents survive. The emergence of a behavioural regularity is interpreted as the emergence of norms. Results reveal that this is not the case. For this reason a further game is introduced, the meta-norms game in which also agents can be punished which do not punish a defector. In simulation experiments this game seemed to suffice for the emergence of norms [However see 16]. This brief example shall highlight the structure of the agents' actions and decision: Agents face three possible actions: first of all agents decide to cooperate or defect. In this setting the norm prescribes to cooperate. Furthermore agents can decide to sanction. Sanctioning is typically modelled in terms of numerical values, i.e. the sanctioned agent looses a certain amount of numerical utility. Thus agents calculate and react to their utility gains. How sanctioning reduces utility values is not made explicit. In consequence agents immediately realise if they have been sanctioned. While recent research on evolutionary game theoretical models show that altruistic behaviour may emerge in certain circumstances even in the absence of sanctions [17,18], also these model do not explain sanctions. The message that can be taken away for the purpose of this paper is that sanctioning is perceived as unambiguous.

Models based in cognitive science apply a more differentiated concept of norms. Rather than being reduced to cooperative behaviour norms can be recognized by agents as a cognitive concept, i.e. it is a deontic belief that people ought to behave in a certain way. In this manner it is possible to model for instance norm innovation as the acquisition of a new deontic belief [19,20]. In both frameworks agents learn norms by being faced with sanctions in correspondence to certain behaviour. In [21,22] the deontic concept of norms is applied for studying the effectiveness of sanctions. Basically a game theoretical setting is extended with a deontic message, informing the victim about the normative reason for the punishment. On a more conceptual level [23] develops a sophisticated process of normative reasoning. [24] develops a cognitive model for differentiating different types of motivation for sanctioning behaviour. Thus, typically the cognitive models concentrate on the reverse side of the process involved in sanctioning: The active side of the punishing agent. If the punished agent is considered as in case of models of normative learning agents need to identify norm violations but not interpret the act of punishment itself. A notable exception can be found in [25] which documents experiments on the likelihood to accept punishments based on desire for others' esteem and to meet others expectations. This is closely related to the question sanction recognition. However, also here others' aggression is equal to punishment

3 EMPIRICAL DATA

In the following an example from research on intra-organizational norms in criminal group will show the ambiguity inherent in the recognition of sanctions. Data had been police files of a criminal investigation. The criminal activities consisted of drug trafficking and laundering the illegal money gained in the drug business. Drug trafficking was undertaken by 'black collar criminals' with access to the production and distribution of drugs. 'White collar criminals' were ordinary businessmen responsible for the money laundering. The psychological techniques applied to draw them in the illegal world beyond a point of no return will not be subject here [26]. Police files identified (at least) one white collar criminal working in the real estate business. It is important that the real estate trader had a good reputation in the legal society. This allowed him to invest illegal money in the legal market and give the return of investment back to the investor, i.e. a black collar criminal. Money laundering is essentially based on a norm of trust: the black collar criminals need to hand over the money to their partners and trust them that they will get the return of investment back from the trustee. In a covert organization this cannot be secured by formal contracts. Therefore trust is essential. The network lasted for about 10 to 15 years until it collapsed. An initial divide went out of control. Mistrust spread rapidly through the whole network, generating a cascading effect through the network which destroyed the overall network in a violent blow-up. Conflicts escalated to a degree of violence that has been described by witnesses as a 'rule of terror' in which 'old friends were killing each other'. In fact, many members of the network were killed. For the individuals involved in the situation, this 'rule of terror' could not be attributed to an individual member any more. Instead, from the perspective of the subjective perception of the group members the terror regime had to be described as governed by an invisible hand. Development of a conceptual model of the data revealed that an ambiguity in recognizing sanctions was of central importance for the collapse.

4 A CONCEPTUAL MODEL

The data is transformed in a conceptual model with a tool denoted as CCD (consistent conceptual description) [27,28]. The CCD tool provides an environment for developing a conceptual model by a controlled identification of condition-action sequences which represent the mechanisms at work in the processes described in the data. Empirical traceability is ensured by tracing the individual sequences back to text annotation in the data. Moreover; the CCD tool creates a code template which can be implemented in a simulation model. The transformation tool called CCD2DRAMS allows the semi-automatic transformation into a basic simulation model that preserves the empirical annotations during the simulation runs. Thus the tool provides a bridge from the evidence base to a simulation [28].

To investigate the particular process of reasoning on aggression in sanction recognition we focus on selected elements of how an initial mistrust generates a positive feedback loop of conflict escalation. How the process starts is displayed in figure 1. It shows an abstract event-action sequence which is derived from the analysis of the data. The box with a red flag represents an event. The action is represented by a box with a yellow flag. Moreover, in bracket we see the possible type of agents that can undertake the action. The arrow represents the relation between the event and the action. This is not a deterministic relation. However, the existence of the condition is necessary for triggering the action. Once an action is performed a new situational condition is created which again triggers new actions.

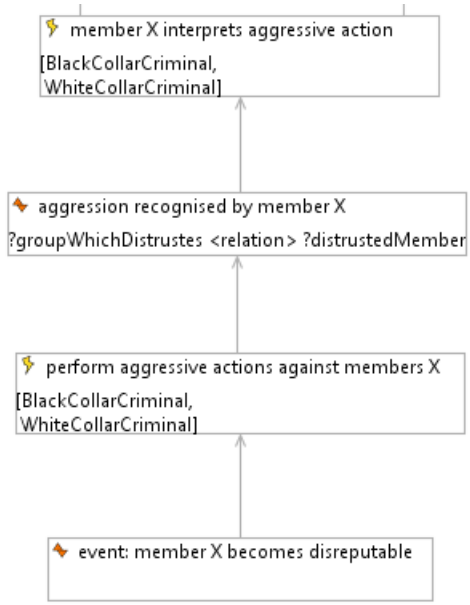


Fig. 1: Initiation of aggression

In the figure, the process starts with the event that someone becomes disreputable which triggers the action of performing an act of aggression against this person. When the victim recognises the aggression, it needs to interpret the motivation. Here two options are considered as possible. This process of reasoning on aggression is displayed in figure 2.

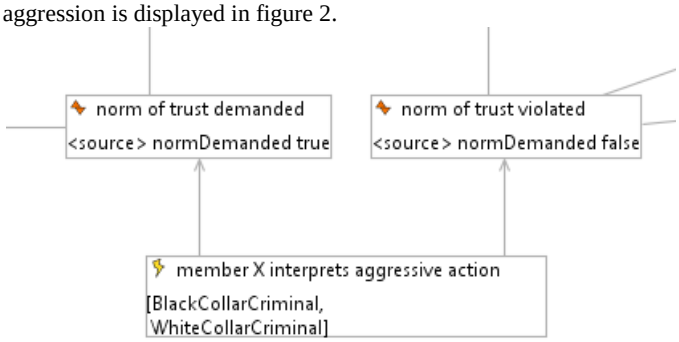


Fig 2: Interpretation of aggression

Figure 2 shows a branching point in the interpretation: The perceived aggression can be interpreted either as norm enforcement, denoted as 'norm of trust demanded', or as norm deviation, denoted as 'norm of trust violated'. Dependent on the interpretation different action possibilities are triggered. We do not go into the details here. However, we show an example how these abstract mechanisms can be traced back to the data. Starting point is the event that for some reasons (out of the scope of the investigation) some member of the organisation becomes distrusted (see figure 1). Empirically the spread of mistrust in the group was initiated by a severe aggression[2]:

Annotation (perform aggressive action against member X): "An attack to the life of M."

It remains unclear who commissioned the assassination and for what reason. It shall be noted that it is possible that an attack on the life could be the execution of a death-penalty for deviant behaviour from his side such as being too greedy. In fact, some years later M. had been killed because he had been accused of stealing drugs. It remains unclear whether this was true or the drugs just got lost for other reasons. However, the murder shows that death penalty is a realistic option in the interpretation of the attack on his life. However, M. survived the attack which allowed him to reason on the aggression. This is the interpretative process displayed in abstract terms in figure 2. No evidence can be found in the data about this reasoning. However, it can be found in the data how he reacted.

Annotation (member X decides to betray criminal organisation): Statement of V01: "M. told the newspapers 'about my role in the network' because he thought that I wanted to kill him to get the money."

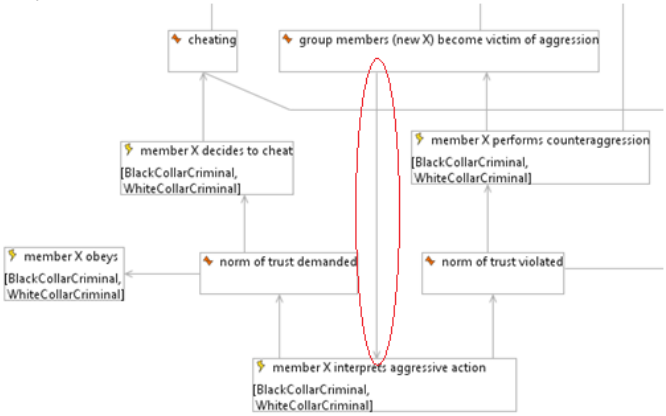


Fig. 3: Instantiation of a feedback loop

This example provides insights into processes of reasoning about aggression: First, he was simply wrong in the assumption that this particular member of the organisation (V01) mandated the attack. Nevertheless it is not completely implausible consideration. M. was one of the black collar criminals who invested money in the legal market though the white collar criminals. V01 was one of the white collar criminals. Thus V01 possessed a considerable amount of drug money which he could have kept for himself if the investor (in this case M.) would be dead. This might be a 'rational' incentive for an assassination. Second, it can be noted that M. interpreted the attack on his life not as a penalty (i.e. death-penalty) for deviant behaviour from his side3. Instead he concluded that the cause of the attack was based on self-interest (the other criminal 'wanted his money'). Thus he interpreted the attack as norm deviation rather than enforcement (see figure 2). Next, he attributed the aggression to an individual person and started a counter-reaction against this particular person by betraying 'his role in the network'. This is an example that he interpreted the aggression as a violation of his trust in V01 and reacted by betraying him. This counter-reaction provoked further panic of other group members such as the one who had been reported here and brought into trouble. Thus his reaction caused further reasoning about the cause of and possible reactions to his aggression. In figure 3 it is indicated that this enfolds a positive feedback loop. Now a new member of the organization undertakes the same reasoning process, whether to interpret the aggression as a violation of a norm of trust

[2] To preserve privacy of data, names have been replaced by notations such as M., V01 etc.

[3] It shall be noted that also the other interpretation in the branching point can be found in the data which is illustrated in the following statement: *Annotation (member X obeys):* "I paid, but I'm alive."

and how to react. Positive feedback cycles may easily become unstable. Thus they are a well known cause for generating strange systemic behaviour. Here it generated a cycle of revenge and counter-revenge which finally went out of control.

5 THEORETICAL ANALYSIS

The empirical case shows that sanction recognition remains ambiguous, if it cannot be secured by the state monopoly of violence. However, what actually are sanctions? In behavioural terms, sanctions belong, next to for instance war, to the very few occasions of socially permitted and even requested aggression. It can be an informal sign such as humping a horn in the case of car drivers, signalling their dissatisfaction with other road users. Aggression can also be highly regulated by legal code and eventual court decisions. Thus sanctioning is a form of regulated aggression among humans. While it may be phenomenological plausible that actors realize that they are victim of an aggression, further inference is necessary for identifying the reasons for the aggression. As it becomes obvious in the case of criminals, not every aggression is norm enforcement. For instance, stealing drugs is a strong temptation for criminals in the drug market. Likewise beating an old lady for stealing her purse is a severe norm deviation. Thus at least two extreme cases can be contrasted:

1. Aggression can be self-interested norm deviation. A hold-up might be one example.
2. Aggression can be socially inspired norm enforcement. Scolding someone for throwing trash just on the ground instead of using a litter bin might be but one example.

This broad distinction is not a fine grained, comprehensive categorization. For instance, aggression might also be an emotional reaction without considering any consequences, be they personally or socially beneficial or not. However, inherent in sanction recognition is an interpretative process. Sanction recognition implies a necessity of reasoning about aggression: namely, interpretation of the motivation. Typically, this is not considered in normative simulation models. This leads to the question whether the emergence of a normative order of a code of procedure can emerge from the scratch. Next, a model of extended normative reasoning is presented.

6 SIMULATING REASONING ON AGGRESSION

The conceptual model has been transformed in an agent-based simulation model. Development of the simulation model has been undertaken in the framework of the code template generated by the CCD tool. The model includes the actor types Black Collar Criminal, White Collar Criminal and Police. Moreover, the general public is included as a static entity.

A comprehensive description of the full model can be found in [29] but is beyond the scope of limits of this paper. For the purpose of demonstrating the extended normative reasoning not the entire model will be presented but rather how the specific normative aspect of sanction recognition is modeled. For this purpose two further cognitive concepts are of fundamental importance: Following [30] criminals are endowed with image and reputation. Both are properties expressing the standing of a criminal, the rank in the hierarchy in a way. Reputation is set for each criminal agent in the initialization of a simulation run, is known to all members of the criminal network and does not change in the course of time. In contrast, the image is information private to each criminal agent, i.e. each criminal agent has its own image of each fellow criminal. Reputation is an objective property of the criminals while image denotes the subjective evaluation of the fellows by each member of the gang. The image of an agent x in the 'mind' of agent y represents the personal experience of the agent y in the interaction with agent x. Reputation results from the common evaluation of an agent by a whole group. For instance an agent y can learn that agent x has a high reputation even though it might not have any personal experience with this agent x. Levels of image and reputation are ordinal scaled attributes: very high, high, modest low and very low. The image values do change during simulation runs: Whereas observation of deviant behavior decreases the image that the observing agent has of deviant agent, observation of acceptance of punishment works in the converse direction and image increases again. If the normative action was a norm violation, then the image strongly decreases ("two levels"), in the case of norm obedience (not shown in the decision tree) the image increases by one "level". Moreover, perception of aggression as norm enforcement increases the image of aggressor in the 'mind' of the victim, whereas perception of aggression as norm violation decreases the image of the aggressor.

The dynamics is modeled in a tick-based way. A tick represents one stage of action or cognition. The model starts with an initial normative event at the first tick regarding a random criminal. This is an unspecified violation of intra-organizational norms that stimulates the necessity of conflict regulation. This normative event is observed by fellow criminals which results in an aggression. The victim experiences the aggression and starts with an interpretation process. An overview of the interpretation process is provided in fig. 4. In fig. 4 x represents the agent that has been selected randomly of being accused of an unspecified norm violation. This is observed by the fellow criminal y which reacts by an aggression. Therefore y needs to interpret the aggression performed by y.

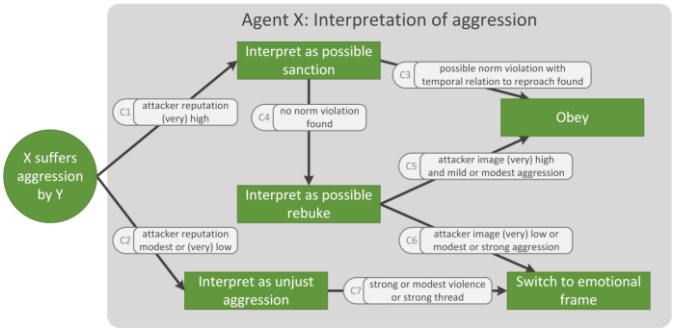


Fig. 4: Interpretation of aggression

The interpretation begins with the distinction whether the aggressor is reputable or not. In the latter case, the aggression is regarded as unjust which triggers an obligatory reaction in the next tick (C2). The agent gets in rage (emotional frame) and reacts by some counter-aggression. If the aggressor is judged to be reputable, then a normative process is performed (C1). Thus only reputable agents are justified to sanction. The normative process can have two results: either it leads to the conclusion that a norm is indeed demanded, persuading the criminal to obey, or that no norm is demanded. The second stage is reasoning about whether the attacked criminal might have violated a norm in the recent past

which would have led to a sanction of another fellow criminal. The basic idea of this normative reasoning is quite simple: It is evaluated whether own actions performed in the past stand in some kind of temporal relationship with a normative event assigned to this criminal. Literally speaking the agent x checks in its memory whether it violated the group norms. In order to conduct this evaluation, each criminal can access a global event board where all aggressions performed by each criminal are recorded. Also the normative events are logged in a similar way, so that temporal relations between these types of events can easily be derived. The normative process is considered successful, if aggressions are found which at most 16 ticks later led to normative events (C3). The 16 ticks represent the length of the memory of the agent [comp. 19 for a similar account]. If such relations exist, the criminal regards a norm demanded and typically react with obeying to the normative request. However, even if the normative process failed (C4), the aggression might still be regarded as a justified sanction: If the attacker has a high or very high image, and the aggression was mild or modest (C5), then it is assumed that a norm is demanded as well. This cognitive heuristic has been included in the model to cover the possible aptitude of criminals with high image (and high reputation) to mitigate conflicts, either by mediating or by just exercising authority. If the aggression is not too severe the simulated criminals do not argue with the bosses.

7 PRELIMINARY SIMULATION RESULTS

During a simulation run the agents develop a personal history of experience with aggressions. These are interpreted differently. In the following screenshots of a particular run will be shown. First figure 5 shows the initialization of a simulation run. The number of reputable agents can be initialized by the user. Image is initialized randomly. Figure 6 shows how the simulation proceeds. Initially one agent has been selected randomly for the initial normative event. Criminal-1 is accused for an unspecified norm violation. This is observed by Reputable Criminal 1 which reacts by an aggressive act, in this case the modest aggression of an ‘outburst of rage’.

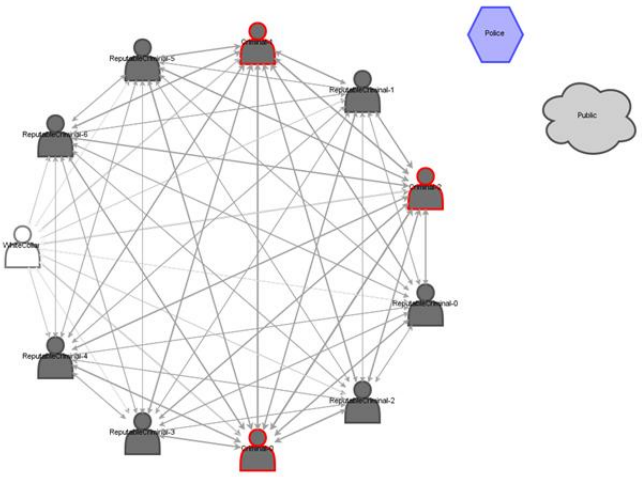


Fig. 5: Initialization of the simulation. The user interface shows the network of criminals, consisting of one white collar criminal (in white), 7 reputable and 3 ‘ordinary’ criminals. The ‘ordinary’ criminals have a red margin to indicate that they are in an emotional frame. The arrows represent the strength of the personal image of the agents. Police might interfere at some instance, represented by a hexagon. Additionally the public is a passive entity represented as a cloud.

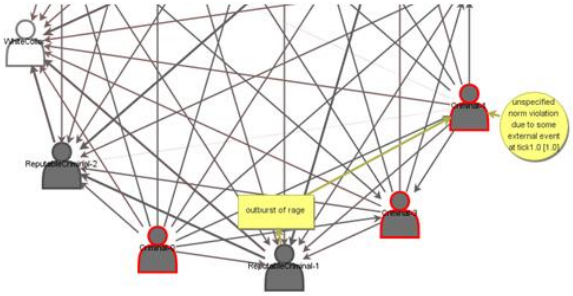


Fig. 6: Reaction on the initial unspecified norm violation of the agent criminal-1 by the agent reputable criminal-1.

As the agent Criminal-1 survives the ‚outburst of rage‘, the agent recognizes the aggression and is able to reason about the aggression. As Reputable Criminal-1 fulfills the condition of being a possible normative authority (because it is a reputable agent), Criminal-1 checks the event board for a possible norm violation on its part. However, the initial event is not stored in the event board and the agent finds no norm demanded. This is shown in figure 7.

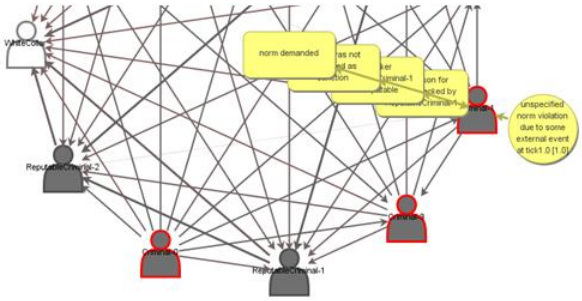


Fig.7: Reasoning on aggression: While the victim of the aggression realizes that the aggressor is reputable and therefore legitimized to potentially sanction norm deviation, the victim does not find a norm violation in the memory. Therefore the agent does not interpret the aggression as sanction.

In the next step however, the agent inspects its personal image of the attacker. At this moment the image is high and for this reason the agent obeys even though it does not find a norm violation. In consequence Criminal-1 updates its image value of Reputable Criminal-1 by increasing its image. Reputable Criminal-1 is becoming a temporary normative authority for Criminal 1. This is shown in figure 8.

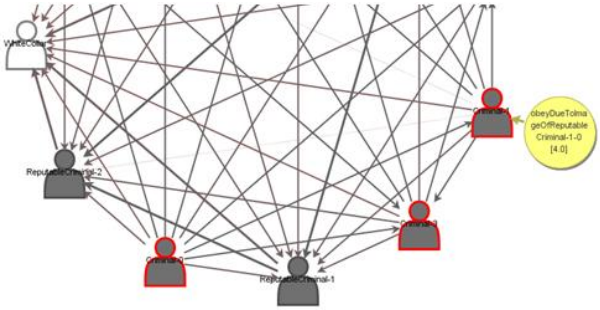


Fig. 8: Final result of the interpretation of aggression. Even though the victim does not find a norm violation, the agent additionally checks its personal image of the aggressor. As the image is high, the agent

subordinates to the authority of the aggressor and does accept the aggression.

Table 1 shows the image values of the agent Criminal-1 in this run, the one randomly selected for the initial normative event. The agent experienced an aggression and interpreted it as sanction due to the image of the aggressor. As this interpretation increases the image of the aggressor it increases the likelihood that further aggression by that agent will be interpreted as sanction again. The agent Reputable Criminal-1 becomes a normative authority for this particular agent.

Table 1. Development of selected image values of agent criminal-1.

Tick	Criminal 6	Reputable Criminal 0	Reputable Criminal 1
1	modest	High	High
2	modest	high	high
3	modest	high	high
4			veryhigh
	modest	high	(sanction)
5	modest	high	veryhigh
6	modest	high	veryhigh
7	modest	high	veryhigh
8	modest	high	veryhigh

8 CONCLUSION AND FUTURE WORK

Sanction recognition suffers from ambiguity. As the consequence recognition of a sanctioning may fail. It is empirically mistaken to perceive sanctioning as a basic theoretical term that needs no explanation. This demonstrates that it is necessary to include sanction recognition in a sociological analysis of norms. This shows that sanction recognition is a black box in normative agent-based models which need to be filled by intra-agent processes of reasoning on aggression that reflects the empirical finding. The gap in recognizing sanctions is overcome by extending normative reasoning by two branching points: first, agents decide whether aggression is performed by a possibly legitimate authority. For this reason reputation of agents is included. Only in that case agents continue reasoning by inspecting an event board whether they undertook an action which could have been a norm violation. If a possible norm violation is found agents react by an act of obedience that reflects a kind of apology. If no such event can be found, a second branching point is included: Agents may still obey if they have a high subjective image of the aggressor. In that case they subordinate to the aggressor by deciding not to dispute the action of the high ranking agent. In both branching points deciding to obey increases the image of the aggressor as a legitimate normative authority.

A comprehensive analysis of the model's behaviour space is still work in progress. For this reason only preliminary hints can be provided whether the emergence of a normative code of conduct is possible from the scratch without social structure that safeguards normative authorities. The emergence implies the development of normative authorities that are legitimized to perform acts of aggression. In the model these are with a high image. Figure 9 shows first results of the development of the average image of the criminals during a selected different simulation run. The average image can have values between +2 (very high) and -2 (very low). The graph shows that the image does not remain stable over time. However, including some ups and downs Reputable Criminal-2 performs best. Nevertheless, authority does not converge to a stable steady state remains fluid during the simulation. Inspection of the subjective image values of the agents shows that image does not converge between the agents.

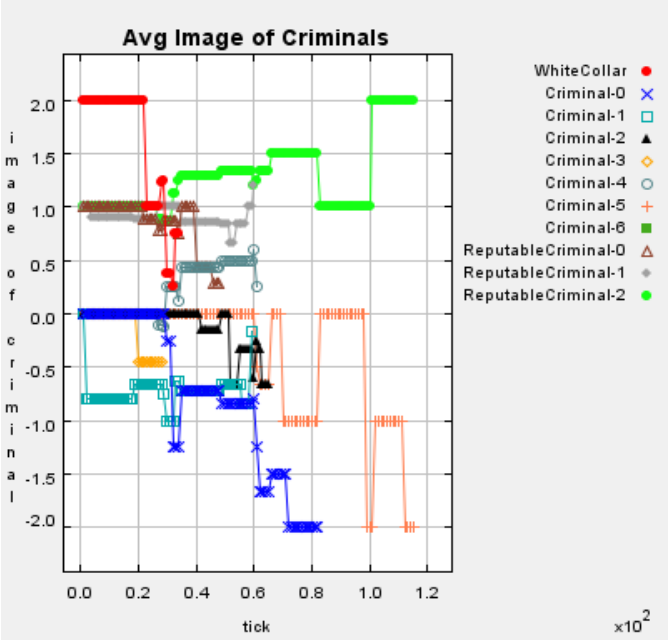


Fig 9: Development of average image values in a selected run.

Thus, simulation runs show the development of a normative authority. However, this is only temporary and subjective. The authority need not be stable, but rather remains fluid and need not converge between the agents but remains ambiguous. For this reason authority may collapse at any time.

ACKNOWLEDGEMENT

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 315874., GLODERS Project.

REFERENCES

[1] O. Barreteau et al., Our Companion Modelling Approach. *Journal of Artificial Societies and Social Simulation* 6 (1), (2003).

[2] G. Lindemann, *Weltzugänge. Die mehrdimensionale Ordnung des Sozialen.* Velbrück, Weilerswist, (2014).

[3] E. Fehr, and S. Gachter, Cooperation and punishment in public goods experiments. *American Economic Review* 90(4), 980-994, (2000).

[4] J. Henrich and R. Boyd, Why people punish defectors. Weak conformist transmission can stabilize costly enforcement of norms in cooperative dilemmas. *Journal of Theoretical Biology* 20(8), 79 – 89, (2001).

[5] J. Henrich, R. McElreath, A. Barr, J. Ensminger, C. Barrett, and A. Bolyanatz, Costly punishment across human societies. *Science*, 312 (5781), 1767-1770, (2006).

[6] C. Bicchieri and R. Muldoon, Social Norms. In: E. N. Zalta (ed.), *The Stanford Encyclopedia of Philosophy*, 2014. URL= <http://plato.stanford.edu/archives/spr2014/entries/social-norms/>.
[7] M. Hechter and K.D. Opp, *Social Norms.* Russell Sage Foundation, New York, 2001.
[8] E. Ullman-Margalit, *The emergence of norms.* Oxford university press, Oxford, 1978.
[9] C. Bicchieri, *The grammar of society. The nature and dynamics of social norms.* Cambridge University Press, New York, 2006.
[10] C. Horne, Explaining norm enforcement. *Rationality and Society* 19(2), 139–170, (2007).
[11] R. Carnap, Testability and meaning pt I. *Philosophy of science* 3: 419-471, Pt II Philosophy of science 4, 2-40, (1936/7).
[12] J. Sneed, The logical structure of mathematical physics. Reidel, Dordrecht, 1971.
[13] V. Gadenne, Theoretische Begriffe und die Prüfbarkeit von Theorien. *Zeitschrift für allgemeine Wissenschaftstheorie* 16(1), 19 – 24(1985).
[14] M. Neumann, Homo socionicus. A case study of simulation models of norms. Journal of Artificial Societies and Social Simulation 11(4), (2008).
[15] R. Axelrod, An evolutionary approach to norms. *American Political Science Review* 80(4), 1095 - 1111, (1986).
[16] M. Galan and L. Izquierdo, Appearances can be deceiving: Lessons learned Re-Implementing Axelrod's 'Evolutionary Approach to Norms'. *Journal of Artificial Societies and Social Simulation* 8(3), (2005).
[17] D. Helbing and W. Yu, The outbreak of cooperation among success-driven individuals under noisy conditions. *Proceedings of the national academy of science* 106(10): 3680-3685, (2009).
[18] D. Helbing and H. Gintis, Homo socialis. An analytical core for sociological theory. *Review of behavioural economics* 2(1-2): 1 – 59, (2015).
[19] T. Savarimuthu, S. Cranefield, M.A. Purvis and M. K. Purvis, Obligation Norm Identification in Agent Societies. *Journal of Artificial Societies and Social Simulation* 13(4), (2010).
[20] R. Conte, G. Andrighetto, M. Campenni (Eds.), *Minding norms. mechanisms and dynamics of social order in agent societies.* Oxford university press, Oxford, (2014).
[21] G. Andrighetto, J. Brandts, R. Conte, J. Sabater-Mir, H. Solaz, and D. Villatoro, Punish and Voice: Punishment Enhances Cooperation when Combined with Norm-Signalling. *PLoS ONE* 8(6).
[22] D. Villatoro, G. Andrighetto, J. Sabater-Mir, and R. Conte, Dynamic sanctioning for robust and cost-efficient norm compliance. *Proceedings of the 22nd international joint conference in artificial intelligence, Barcelona, Spain, July 16-22*, (2011).
[23] C. Hollander and A. Wu, The Current State of Normative Agent-Based Systems. *Journal of Artificial Societies and Social Simulation* 14(2), (2011).
[24] F. Giardini, G. Andrighetto, R. Conte. A cognitive model of punishment. Proceedings of the 32nd annual conference of the cognitive science society, 11-14 August 2010, 1282-1288, (2010).
[25] G. Andrighetto, D. Grieco, L. Tummolini, Perceived legitimacy of normative expectations motivates compliance with social norms when nobody is watching. Frontiers in Psychology 6: 1413
[26] C. van Putten, The process of extortion: problems and qualifications. *Conference on extortion racket systems*. University of Vienna, Vienna, 7 – 11, 2012.
[27] S. Scherer, M., Wimmer, and S. Markisic, Bridging narrative scenario texts and formal policy modelling through conceptual policy modelling. *Artificial Intelligence and Law* 21(4), 455 – 484, (2013).
[28] S. Scherer, M. Wimmer, U. Lotzmann, S. Moss, and D. Pinotti, An evidence-based and conceptual model-driven approach for agent-based policy modelling. *Journal of Artificial Societies and Social Simulation* 18(3), (2015).
[29] U. Lotzmann and M. Neumann, A simulation model of intra-organizational conflict regulation in the crime world. In C. Elsenbroich, D. Anzola, and N. Gilbert (eds.), *Social dimensions of organized crime*, Springer, New York, 222-262, (2016).
[30] J. Sabater-Mir, M. Paolucci and R. Conte, Repage: REputation and imAGE among limited autonomous partners. *Journal of artificial societies and social simulation* 9(2), (2006).

An Architecture for the Legal Systems of Compliance-Critical Agent Societies

Antônio Carlos da Rocha Costa [1]

Abstract. This paper introduces an architecture for the legal systems of agent societies. The reasons for an agent society requiring the constitution of a legal system of its own are exposed. The main features of Hans Kelsen's concept of legal system are reviewed. The way those features determine the proposed architecture is explained. A brief case study is presented, to help to contextualize and make concrete the ideas of the paper. Finally, the concept of compliance-critical agent society is introduced and related to the body of the paper.

1 Introduction

This paper adopts a particular concept of agent society, and introduces an architecture for legal systems constituted in agent societies that are conceived in accordance with that concept. The main features of such architecture were derived from an operational reading of Hans Kelsens theory of legal systems [20] which, differently from Ronald Dworkin's [11] or H. L. A. Hart's [13] theories, focuses on the structural and dynamical aspects of legal systems, not on their contents.

The paper is structured as follows. Section 2 presents a view of the historical evolution of multiagent systems, situating *agent societies* in the current evolution stage, detaching them from the *agent organizations*, which were the focus of the previous stage. Section 3 briefly analyzes the notion of *entanglement* of agent and human societies, that seems to be emerging from the current stage of evolution.

Section 4 reviews in formal terms the concept of agent society adopted in the work.

Section 5 discusses the main types of situations in which an agent society may have to constitute a legal system of its own, to become capable of adequately regulating its structure and functioning. Section 6 summarizes the operational reading of Kelsen's theory of legal systems presented in [7].

Section 7 reviews the formal concept of legal system of agent society that emerged from that reading of Kelsen's theory and sketches the proposed architecture of legal systems of agent societies.

Section 8 presents a sample application, sketching a model for the way legal systems may support public policies. The section is intended as an illustration of the way agent societies endowed with legal systems may be used as semantical bases for the modeling of legal issues in public policy simulation efforts.

[1] Programa de Pós-Graduação em Informática na Educação da UFRGS. 90.040-060 Porto Alegre, Brazil. Programa de Pós-Graduação em Computação da FURG. 96.203-900 Rio Grande, Brazil. Email: ac.rocha.costa@gmail.com .

Section 9 discusses some of the issues raised by the paper and comments related work. Section 10, presents the main conclusion of the paper, namely, that agent societies that are entangled with human societies may have to be taken as a particular type of critical systems, and have their development subject to the requirements that the development of critical systems usually has.

2 The Historical Evolution of Multiagent Systems

Figure 1 illustrates our view of the historical evolution of multiagent systems.

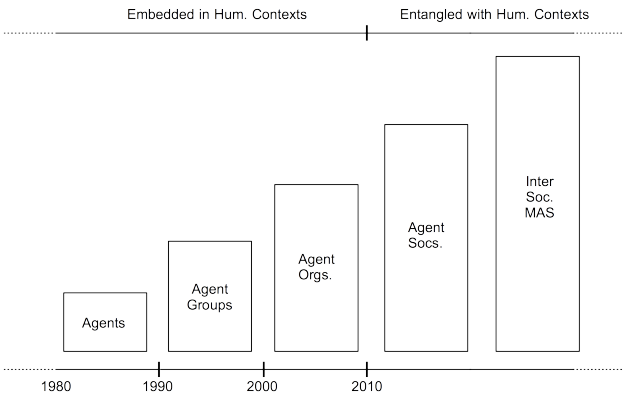


Figure 1. A view of the temporal evolution of multiagent systems.

We characterize each stage as follows:

- in the first stage (*Agents*, up to the 1980's), the focus was on the development of agents and their interactions: a representative model is, e.g., in [22];
- in the second stage (*Agent Groups*, in the 1990's), the focus was on the development of agent groups and the roles agents played in groups: a representative model is, e.g., in [12];
- in the third stage (*Agent Organizations*, in the 2000's), the focus was on the development of agent organizations and the systems of norms that regulate their structure and functioning: a representative model is, e.g., in [18];

- we submit here that we are facing now the fourth stage, and the eve of the fifth one (*Agent Societies* and *Inter-Societal Multiagent Systems*, from the 2010's on), where the focus is in the development of full-fledged agent societies (with social sub-systems constituted by inter-organizational structures) and of systems of multiple agent societies (structured on the bases of inter-societal exchanges).

We claim that the model of agent society adopted in the present work (see Sect. 4) is representative of the focus of the fourth stage.

3 Agent Societies and their Entanglement with Human Societies

We remark that the first three stages of the historical evolution discussed above conceived multiagent systems as systems *embedded* in human contexts, that is, multiagent systems were conceived as interacting with individual users or groups of users, on the basis of an interface that encapsulated inside the multiagent system all the structures and functional processes that it required.

Agent societies and inter-societal multiagent systems, on the other hand, seem to tend to operate in a way that *entangles* their structure and functioning with the structure and functioning of the human contexts where they are situated [2].

That is, their tendency seems to further the embedding of multiagent systems in human societies, by leading the individual users or groups of users of a multiagent system to participate within that multiagent system, by becoming structurally involved (directly or through avatars) in the performance of its functional processes [3].

As argued in [6], such type of entanglement opens the way for the rising of legal and moral implications for the design and operation of agent societies.

The present paper aims to contribute to the treatment of the legal issues that may arise from such entanglements, by presenting an architecture for legal systems of agent societies, that is, full-fledged legal systems constituted and operated internally to agent societies. Section 5 discusses in more detail the reasons for the constitution of such legal systems.

4 A Formal Concept of Agent Society

We have been using in our work a concept of agent society aiming at a full-fledged social structure and functionality, to meet the requirements of the current stage of the historical evolution of multiagent systems that we identified above.

For that, we define an *agent society* as a particular type of multiagent system endowed with the following characteristics:

- *openess*, meaning that the agents can freely enter and leave the system;
- *organized*, meaning that the functioning of the society is given by a system of processes for which it is possible to separate *individual processes*, performed by single agents, and *social processes*, performed by sets of agents (said to be the organizational units of the society), but so that for each agent can determine the part it plays in each social process (possibly, none);
- *persistent*, in the sense that the system of social processes of the society, and the organizational units that perform them, persist in time, independently of which individual agent enters or leave the society (up to a minimum population of agents in the society);
- *situated*, in the sense that the society exists and operates in an environment (material and/or symbolic), whose objects may be taken by the agents and organizational units to help them to perform their individual and social processes.

Figure 2 sketches the architecture that we envisage for agent societies, emphasizing the various levels of their organizational structure:

- the *micro-organizational* level (Org_ω), constituted by the *organizational roles* (Ro) that the individual agents may play in the society and in the organizational units;
- the *meso-organizational* level (Org_μ), constituted by the *organizational units* (OU), each constituted by a set of organizational roles, structured by a network of role interactions, and each organizational unit possibly recursively based on lower-level organizational units; we call *agent organizations* the maximal organizational units, that is, the organizational units that are not constitutive of higher-level organizational units;
- the *macro-organizational* level (Org_Ω), constituted by *social sub-systems* (SS), each social sub-system constituted by a set of agent organizations, structured by a network of inter-organizational interactions.

Let T denote the time structure. We formally define an agent society as follows (see [4] and [10] for further details):

An *agent society* is a time-indexed structure $AgSoc^t = (Pop^t, Org^t, MEnv^t, SEnv^t, Imp^t, Use^t)$ where:

- Pop^t is the *populational structure* of the society at time t, that is, the network of interacting agents that inhabit it at that time;
- $Org^t = (Org^t_\omega, Org^t_m u, Org^t_\Omega)$ is the *organizational structure* of the society at time t, with:
  - Org^t_ω is the *micro-organizational structure*, that is, the network of interacting *organizational roles* implemented by the population at time t;
  - Org^t_μ is the *meso-organizational structure*, that is, the network of interacting *organizational units* implemented by the organizational roles at time t;
  - Org^t_Ω is the *macro-organizational structure*, that is, the network of interacting *social sub-systems* implemented by the agent organizations of the society at time t;
  - $MEnv^t$ is the *material environment* of the society, constituted by the material objects that the agents and agent organizations may make use of;
  - $SEnv^t$ is the *symbolic environment* of the society, constituted by the symbolic objects that the agents

[2] Entanglements which, due to the openness of agent societies, may acquire dynamical features, both structurally and functionally, perhaps on the basis of modular components of agent societies [5].

[3] And, through the *mutual recursive* entanglement of agent societies with human societies, agents and agent organizations, and individual users and groups of users participate in all the agent and human societies which are mutually entangled.

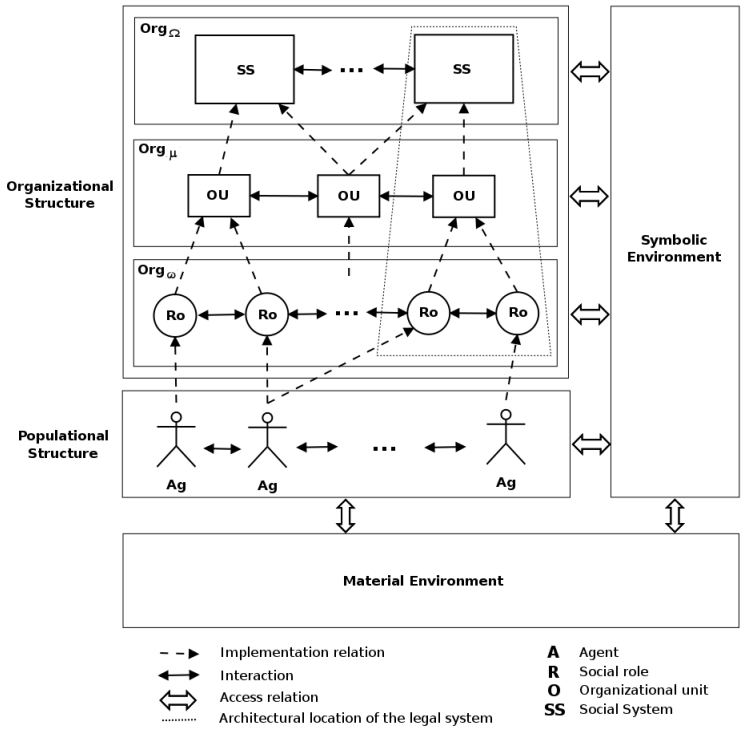


Figure 2. Sketch of a structural model for agent societies, with the location of the organizational part of its legal system.

and agent organizations may make use of [4];

- Imp^t is the *implementation relation*, determining how each organizational level of the organizational structure (the components, their behaviors and their interaction processes) is implemented by the lower one, at time t (including how the micro-organizational structure is implemented by the populational structure);
- Use^t is the *use relation*, determining how the objects of the material and the symbolic environments are used by the components of the populational and the organizational structure, at time t.

Figure 2 illustrates also the locus that a legal system would take in the architecture of the agent society: it should expand the whole range of organizational levels, from the level of the social roles to the level of the social sub-system that is constitutes, passing through the level of the organizational units that constitute a legal system (for instance, law monitors and law enforcers).

It illustrates also that legal systems should have connections with other social sub-systems (for instance, administrative or cultural sub-systems) and that the social roles of legal systems may be performed by agents that also perform other social roles, in other social sub-systems of the society.

[4] The *symbolic environment* is of particular importance for the present work, for it is there that the public symbolic components of the legal system of the agent society are realized (see Sect. 7).

5 Three Reasons for Constituting Legal Systems in Agent Societies

Agent and human societies may entangle to each other in complicated ways. An elementary example of such entanglement, enough to hint on the problems that may arise in more contrived ones, is shown in Fig. 3.

The figure illustrates the agent and human societies, their legal environments (combinations of relevant legal systems), the interactions between agents and humans, and the bindings to the agents and humans to some of the legal environments that are relevant for their actions.

In view of the possibility of such complicated articulations, it seems that there are at least two main reasons for having agent societies equipped with legal systems of their own.

1. *Dynamicity of the legal systems present in a legal environment, and the possibility of the dynamic emergence of legal conflicts between them*:
   Agents and agent organizations should have access to the legal norms under which they operate, should be warned when they violate them, and should informed of the corresponding sanctions, in such cases. All that should be dynamically provided, at run time, to the agents and agent organizations, specially when the legal systems of the legal environment in which they operate are dynamically subject to changes, so that legal conflicts between them may dynamically arise.
2. *Contrived nesting of legal systems in a legal environment*:
   As agent and human societies operate in an articulated way, with agents and agent organizations of the former, and in-

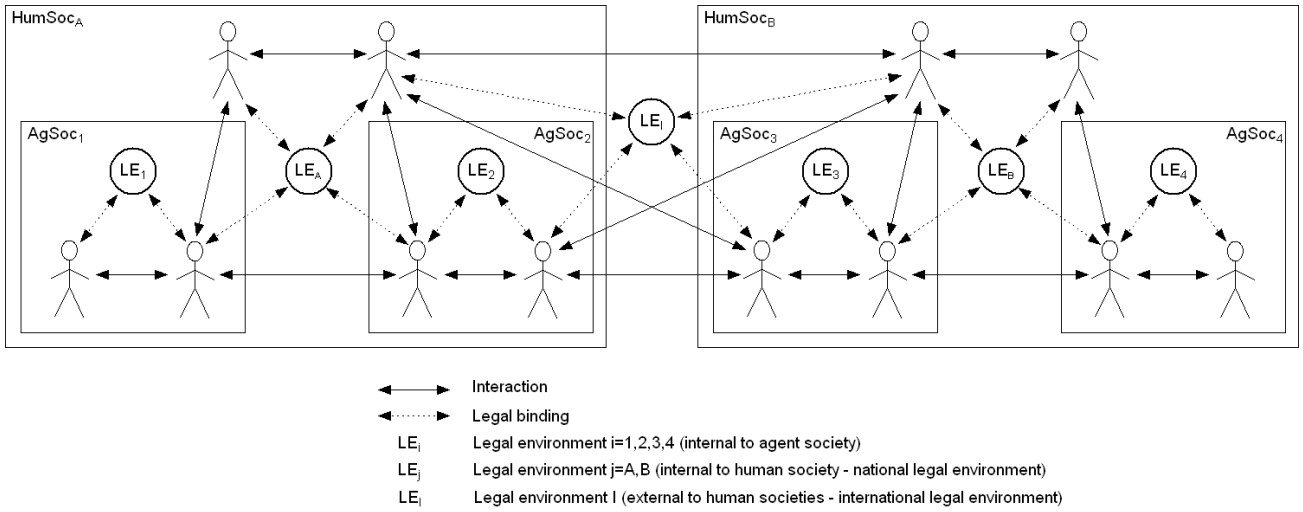


Figure 3. An elementary example of entanglement of agent and human societies, and of their legal systems.

dividual and organizational users of the later, performing systemic functions for each other's societies, the legal systems of agent and human societies may nest to each other in contrived ways, possibly with a variety of legal conflicts among them, it is important that reliable monitoring of conducts, judgments and sanctioning be provided, which can only be made locally, to each agent society.

3. *The need of official record of norm violations and sanction applications, and of their official communication to the legal systems of human societies*:
   Since the decisions and actions of the agents and agent organizations are often encapsulated inside the agent societies where they operate, and (either by demands of efficiency or by security reasons) not always made dynamically available outside those societies by the agents and agent organizations themselves, it is important that official information about violations of legal norms by agents or agent organizations, and official information about the application of sanctions to them, be made available to the legal systems of the human societies to which those agent societies are articulated. This is specially relevant when agent and human societies are entangled, with agents and agent organizations of agent societies operating on behalf for individual or organizational users of human societies, and when individual or organizational users of human societies perform systemic functions for the agents or agent organizations of the agent societies.

We submit that only legal systems officially constituted inside agent societies and appropriately certified by the legal systems of the human societies to which they are entangled (thus operating only under norms and actions legally accepted by those human societies) can satisfy those three requirements.

6 An Operational Reading of Kelsen's Theory of Legal Systems

We summarize here the operational reading of Hans Kelsen's theory of legal systems [20] that was presented in [7].

6.1 Kelsen's Notion of Legal System

Law is, for Kelsen [20], a social technique, that is, a technique of social control. And legal systems are the way they operate inside societies.

The legal system of a society is, thus, a part of that society, one of its *sub-systems*, in the conceptual framework of the model presented in Sect. 4. As such, their basic components are agents and sets of agents constituted as organizational units.

Kelsen calls *legal organs* the agents and organizational units that comprise a legal system. They are constituted as legal organs by the legal system itself, and only when behave and interact in accordance to their legal authorizations they are acting as legal organs.

Kelsen's notion of a legal system, then, is that of a social mechanism [14], one of the mechanisms responsible for controlling the society where it operates.

6.2 Differentiating Legal and Moral Systems

In adopting Kelsen's theory of legal systems in the present work, we adopt also the distinction he makes between legal and moral systems [19], which is important given the importance that moral systems constituted inside agent societies may happen to acquire (see, e.g. [8]).

In particular, we adopt the main implication of that distinction, formulated in [7], namely, that the usual notion of normative system, founded on deontic logic, is too general and abstract to be able to capture the important operational differences between those two specific types of normative systems.

For, the essential difference between legal and moral systems [19] is neither in their possible contents, nor in their logical form, but in the way they are operationalized in a society, specially the mechanism through which norm violations are sanctioned: in legal systems, the sanctioning mechanisms are officially constituted by the legal system itself, often under

a hierarchical organization; in moral systems, the sanctioning mechanisms are based on the subjects of the moral systems themselves, which are personally responsible for sanctioning each other.

6.3 Kelsen's Notion of Legal Norm

In its most general form, Kelsen's conception [20], a legal norm is a statement that if a certain conduct α is performed, another conduct β ought to be performed, as consequence of the performance of the former. The conduct α is called the *condition* of the legal norm and the conduct β, the *sanction* of the legal norm.

Notice, however, that the consequence relation that constitutes a legal norm is not, from Kelsen's point of view, a logical relation, because conducts are not propositions: it is a relation between two actions, two conducts. Kelsen calls it the *imputation relation* [20], and in [7] we denote it by: $\alpha \Downarrow \beta$.

Of course, the execution of conducts can be expressed through propositions, and the imputation relation can be presented in a deontical form: if $\langle\alpha\rangle$ and $\langle\beta\rangle$ are propositions, each denoting the occurrence of its respective conduct, and if $\Box X$ denotes that the obligation of bringing about that X, then any imputation might perhaps be logically expressed by $\langle\alpha\rangle \Rightarrow \Box\langle\beta\rangle$. But that, of course, would obscure Kelsen's emphasis on the operational sense of imputation.

The relation between the two expressions, the operational and the logical, allows Kelsen to distinguish [20] between the *legal language* of a legal system, to which imputations belong, and the *theoretical language* of the jurisprudence about that legal system, to which belong the logical and epistemological propositions that concern that legal system.

That distinction justifies the operational reading of Kelsen's theory of legal system introduced in [7], which provides an operational semantical model for legal systems that sees a legal system as an interpreting mechanism for its legal language.

Imputations, however, are not enough to constitute the basic operational model of legal norms. Besides imputations, the legal languages of legal systems should allow for the notion of authorization: an *authorization* is a conduct that brings about that certain subject of the legal system becomes authorized to perform some conduct.

We denote an authorization stating that a legal subject ag is authorized to perform a conduct β by:

$$Auth(ag \triangleright \beta)$$

However, a full-fledged operational model for legal norms requires the notion that occurrences of conducts be individualized, in the sense at a minimum the occurrence of a conduct should be related to the subject of the legal system that realized it. Thus, instead of denoting the occurrence of a conduct simply by, e.g., α, one is required to denote it by, e.g., $ag \triangleright \alpha$, where ag identifies the legal subject that realized the conduct.

With individualized imputations and authorizations, legal norms can then be given the general operational form [7]:

$$(ag_1 \triangleright \alpha) \Downarrow Auth(ag_2 \triangleright \beta)$$

meaning that, in the legal system where that norm is valid, the realization of the conduct α by the legal subject ag_1 has as a consequence that a legal subject ag_2 ought to be authorized to realize the conduct β (usually, as a way to sanction the realization of α by ag_1).

A variety of decorations (time, location, etc.) can be added to the basic imputation form of legal norms, to enrich it with information specific to a given situation. Also, contextual conditions can be added to conducts, both in the conditions and in the consequences of imputations, to increase the complexity of the legal norms that formal imputations can capture.

This way of formally presenting the concept of legal norm makes explicit that legal norms are operational elements of systems (the *legal systems*), where agents are present (as *legal subjects*), which are put into operation on the basis of the *conducts* of the agents, and whose fundamental semantical model has a dynamical, not of a logical, character.

7 The Proposed Architecture for Legal Systems of Agent Societies

We base the architecture of agent societies proposed here on an operational reading of Kelsen's theory of legal systems. Briefly, the result of such operational reading is the following[5].

7.1 Legal Systems of Agent Societies, Formally Defined

Let $AgSoc^t = (Pop^t, Org^t, MEnv^t, SEnv^t, Imp^t, Use^t)$ be the time indexed-structure of the agent society $AgSoc$. The *legal system* of $AgSoc$ is a time-indexed structure:

$$LSys_{AgSoc} = (\{LOrd^t\}_{t\in T}, \{LOrg^t\}_{t\in T}, \{RLFct^t\}_{t\in T}, createlnrm, deroglnrm, createlauth, cancellauth, recordlfct, deletelfct)$$

where:

- $LOrd^t$ is the *legal order* (the set of legal norms officially acknowledged as valid by the legal system) at time t;
- $LOrg^t$ is the *system of legal organs* (the set of agents and agent organizations officially operating on behalf of the legal system) at time t;
- $RLFct^t$ is the *record of legal facts* (the set of social facts officially accepted as such by the legal system) at time t;
- and the operations are:
  - *createlnrm*, of creation of legal norms, which includes a new legal norm in $LOrd^t$;
  - *deroglnrm*, of derogation of legal norms, which excludes a legal norm present in $LOrd^t$;
  - *createlauth*, of creation of legal authorizations, which includes a new legal authorization in $RLFct^t$;
  - *cancellauth*, of cancellation of legal authorizations, which excludes a legal authorization present in $RLFct^t$;
  - *recordlfct*, of recording of legal facts, which includes a new legal fact in $RLFct^t$;
  - *deletelfct*, of deletion of legal facts, which excludes a legal fact present in $RLFct^t$.

[5] See [7] for the full account of the resulting operational semantical model of legal systems.

7.2 The Proposed Architecture

The formal account of Kelsen's concept of legal system, presented above, motivates the architecture of legal systems of agent societies pictured in Figs. 4 and 5.

Figure 4 shows the location of the the three main public symbolic components of legal systems, namely, the legal order (*LOrd*), the record of legal facts (*RLFct*) and the current set of legal demands (*LDmd*, a subset of *RLFct*).

Given the public character of those symbolic components, they are taken to be stored in the *symbolic environment* of the agent society, so that they can be accessed by every agent or agent organization. The meaning of the arrows is the same as in Fig. 5.

Figure 5, on its turn, gives a schematic view of the main components of the internal organization of a legal system *LSys*. The figure shows the three main *types of legal organs* of the legal system:

1. the *conduct monitor* (*CMon*), responsible for monitoring the conducts (behaviors and interactions) of the agent organizations, and of the agents while performing social roles in the society;
2. the *sanction executor* (*SExc*), responsible for executing the sanctions that should be applied to the agent organizations and to the agents, for the behaviors and interactions that they performed, and which dit not complied with the law;
3. the *norm issuer* (*NIssr*), responsible for creating new legal norms and abrogating existent ones (either general legal norms or individual ones), in response to the legal demands presented by agents or agent organizations, when acting as legal organs validly authorized to place such demands.

Figure 5 also shows the access rights those types of legal organs have on the three main public symbolic components of the legal system: either *reading* or *reading-and-writing*.

Notice how, in general, agent organizations as well as individual agents implementing legal roles in the society have reading-and-writing access right to the current set of legal demands, but only reading access right to the legal order.

Notice, also, that any of the above three *types* legal organs may be instantiated either as a simple legal role (for instance, a judge acting as a law emitter) or as an organizational unit (for instance, a parliament as a law emitter). The only requirement for the validity of those instantiations is that they be validly authorized by the legal system itself.

The figure also shows that legal systems are concerned essentially with the organizational structure of the agent society (the roles that the individual agents perform in agent society, and with the organizational units they implement), not with its populational structure (not with the agents themselves).

This should be understood clearly: to be a *subject* of a legal system is already to perform a *legal role* in the society (two common types of *subjects* of legal systems are, for instance, the *citizens* and the *foreigners* of the society). This is so because it is possible that the legal system of the society restricts its scope of application to just a subset of the population that inhabit the area in which the legal system is considered to be valid, leaving the rest of that population at the margin of the legal system (for instance, treating them as "things" or "animals", which can even be owned by the subjects of the legal system).

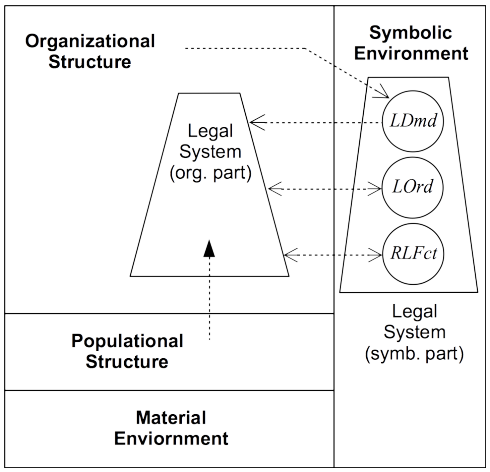


Figure 4. The locations of the organizational part (system of legal organs) and of the symbolic part (the public symbolic components) of the legal system of an agent society.

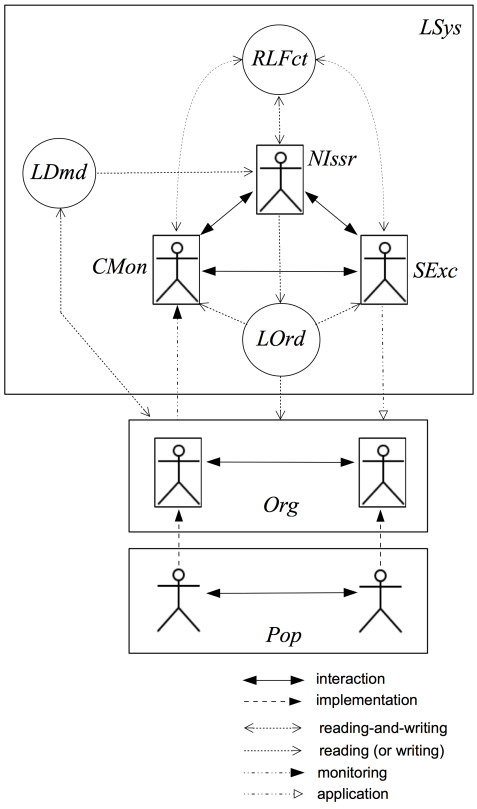


Figure 5. Conceptual model *LSys* of the proposed architecture of legal systems of agent societies (which is partly included in *Org* and partly in *SEnv*, see Fig. 4).

Notice, on the other hand, that the organizational part of the legal system *LSys* (its *system of legal organs*) is part of the organizational structure *Org* of the society on which it operates , as illustrated in Fig. 2.

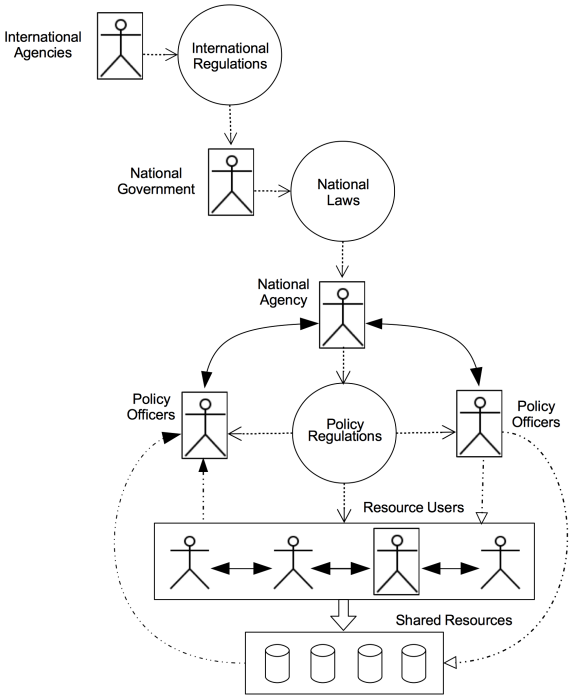


Figure 6. Sketch of a simulation model for the execution phase of a public policy, and for the legal system that supports it.

8 A Sample Application: Modeling Public Policy Systems and the Legal Systems that Support Them

In [9], an agent-based model for the simulation of the execution phase of public policies was introduced, taking as a basis a simple *sequential model* for the *policy cycle* of *public policies* that concerned with the adequate use of *shared public resources* [15].

A case study was conducted concerning the modeling and simulation of public policies for controlling fishing activity during the *Piracema* (fish reproduction period) in Brazilian rivers. Fish populations were treated as resources for common use by fishermen and fish industry.

Fig. 6 shows, in a generic form, the various types of legal organs involved in the formulation and operation of the public policy considered in the simulation model [9]. The arrows are like in Fig. 2.

The *public policy*, directed toward the management of the *shared public resources*, and made positive in the form of a set of *national laws* and *policy regulations* (i.e., administrative plans and norms), is jointly formulated by the *National Government* and the *National Agency* that is responsible for those shared resources.

In formulating *policy regulations*, the *National Agency* has to abide to *national laws* (issued by the *National Government*) and to *international regulations* (issued by the *International Agencies* concerned with that type of resource).

Policy officers operate the public policy by following the policy regulations issued by the National Agency, applying it to the *resource users*, according to the ways they use the resources.

One can easily picture some of the *legal environments* (combined sets of legal systems [6]) involved in the model:

1. the set of *international regulations* constitute an *external legal environment* for the action of the *national government*; it serves as a basis for the constitution of an *internal legal environment*, constraining the *national laws* that the national government can issue;
2. an issued *policy regulation*, on the basis of the national laws, constitutes a *sectorial legal environment* for the resource users;
3. the *policy regulation* and the national laws also constitute a *sectorial legal environment* for the joint operation of the national agency and its officers.

One can immediately see another possible application for such model: it could be embedded in a decision support system used by the politicians and administrators involved in the problem of managing a shared public resource.

The model would be able to support, regarding legal issues, the live participation of users in the simulation of some of the social actors (individual agents, agent organizations, legal organs) present in the situation.

Legal decisions concerning the situation could be essayed, on the basis of such "participatory simulations". That would evince the importance of having an explicit, sound, and complete embedding in the system of the legal aspects at stake: the decision system would be able to immediately indicate the legal consequences, for the simulated situation, of the actions that the users realize in the simulation.

9 Discussion

As already mentioned, this paper is based on operational reading of Kelsen's theory of legal systems, introduced in [7]. Drawing the architectural model of legal systems from such source immediately construed it as an operational semantical model, directly bound to the model of agent societies.

These two features (the operational character and the binding to the adopted model of agent societies) serve to contrast the proposed architectural model from the models of normative systems usually adopted in the current approaches (see, e.g., [1, 2]) to the issue of norms in multiagent systems:

- the adoption of a full-fledged model of agent societies, with each agent society endowed with a full-fledged legal system of its own;
- the founding of the model on an operational semantical account of the structure and functioning of the legal systems, instead of a deontic logical approach, mostly limited to the legal orders of the legal systems;
- the binding of the legal systems to the organizational structure of their underlying agent societies (organizational roles, agent organizations, etc.) instead of their direct binding to the agents of the societies.

In addition, the analysis of the conditions of structural and functional entanglement of agent and human societies points to the particular importance of the internal constitution of full-fledged legal systems in *norm-critical agent societies*, as conceptualized below, in the Conclusion.

10 Conclusion: Compliance-Critical Agent Societies

This paper takes as given that agent societies whose actions and decisions can have legal and moral consequences for the human societies with they are entangled should have their actions and decisions verified and accredited by the legal systems of those human societies.

The paper submits that the most efficient, effective and secure way of providing that verification and accreditation, in view of the openness of the agent societies, is through the verification and accreditation of legal systems constituted in those agent societies. That should guarantee also the correctness and legal validity of the information about those actions and decisions, provided by the legal systems of the agent societies to the legal systems of the human societies.

The architecture for legal systems of agent societies proposed here aims at easing the constitution of such verified and accredited legal systems. It supports the entanglement of agent societies wit a multiplicity of human societies and their varied legal systems, including the international law, which is important in what concerns the international commerce.

Finally, we submit that agent societies endowed with legal systems of their own should be taken as a specific form of *critical systems* (see, e.g., [16]), even if their operation cannot result in physical damages, risks for human life, or economic losses: legal, moral, political, or other types of *social damages* should be enough to warrant that classification.

In consequence, it seems sensible to require that the development of agent societies endowed with legal systems of their own, operating as a specific type of *compliance-critical agent societies*, be subject to the formal methodological care that is usually given to the other types of critical systems.

How far, however, is the current proposal from the available multiagent systems platforms? Not too far, we think. For instance, the Agents & Artifacts model [21], as supported by the JaCaMo platform [3], has proved to be enough not only to support artifacts reifying organizational units [17], but also artifacts reifying components of legal systems [9].

As far as we know, no full-fledge legal system has been defined and implemented for agent societies. Such a work requires, however, not only the adoption of an architectural model for the legal system, as the one we have proposed here. It requires also the detailed specification of the contents of the legal norms of the legal system, of the rules of practice of its legal organs, and of the ways the legal norms and the practices of the legal organs interact with the legal norms and legal practices of the human society to which the agent society is to be entangled. That, of course, is a type of work different from the architectural work that we have presented here.

Acknowledgments

The author thanks the anonymous reviewers for their very useful remarks.

REFERENCES

[1] Guido Boella, Leendert van der Torre, and Harko Verhagen, 'Introduction to normative multiagent systems', *Computational and Mathematical Organization Theory*, **12**, 71–79, (2006).

[2] Guido Boella, Leendert van der Torre, and Harko Verhagen, 'Introduction to the special issue on normative multiagent systems', *Autonomous Agents and Multiagent Systems*, **17**, 1–10, (2008).

[3] Olivier Boissier, Rafael Bordini, Jomi Fred Hübner, Alessandro Ricci, and Andrea Santi, 'Multi-agent oriented programming with JaCaMo', *Science of Computer Programming*, (2011).

[4] Antônio Carlos Rocha Costa. On the bases of an architectural style for agent societies: Concept and core operational structure. *Open publication on* www.ResearchGate.net - DOI: 10.13140/2.1.4583.8720, 2014.

[5] Antônio Carlos Rocha Costa, 'Proposal for a notion of modularity in multiagent systems', in *Informal Proceedings of EMAS 2014*, eds., M. Birna van Riemskijk, Fabiano Dalpiaz, and Jürgen Dix. AAMAS @ Paris, (2014).

[6] Antônio Carlos Rocha Costa. On the legal aspects of agent societies. *Open publication on* www.ResearchGate.net - DOI: 10.13140/2.1.4345.7923, 2014.

[7] Antônio Carlos Rocha Costa, 'Situated legal systems and their operational semantics', *Artificial Intelligence & Law*, **43**(1), 43–102, (2015).

[8] Antônio Carlos Rocha Costa, 'Moral systems of agent societies: Some elements for their analysis and design', in *Proc. EDIA16 - Workshop on Ethics in the Design of Intelligent Agents*, The Hague, (2016). ECAI 2016.

[9] Antônio Carlos Rocha Costa and Iverton Adão da Silva dos Santos, 'Toward a framework for simulating agent-based models of public policy processes on the Jason-CArtAgO platform', in *AMPLE 2012 - 2nd International Workshop on Agent-based Modeling for Policy Engineering*, Montpellier, (2012). ECAI 2012.

[10] Antônio Carlos Rocha Costa and Graçaliz Pereira Dimuro, 'A minimal dynamical organization model', in *Hanbook of Multi-Agent Systems: Semantics and Dynamics of Organizational Models*, ed., V. Dignum, 419–445, IGI Global, Hershey, (2009).

[11] Ronald Dworkin, *Taking Rights Seriously*, Harvard Univ. Press, 1977.

[12] J. Ferber and O Gutknecht, 'Aalaadin: a meta-model for the analysis and design of organizations in multi-agent systems', in *International Conference on Multi-Agent Systems - ICMAS 98*, ed., Y. Demazeau, pp. 128–135, Paris, (1998). IEEE Press.

[13] H. L. A. Hart, *The Concept of Law*, Oxford University Press, 2012.

[14] *Social Mechanisms. An Analytical Approach to Social Theory*, eds., Peter Hedström and Richard Swedberg, Cambridge Univ. Press, Cambridge, 1998.

[15] Michael Hill, *The Public Policy Process*, Pearson Longman, London, 2009. (5th ed.).

[16] Cris Hobbs, *Embedded Software Development for Safety-Critical Systems*, CRC Press, Boca Raton, 2015.

[17] Jomi F. Hübner, Olivier Boissier, R. Kitio, and Alessandro Ricci, 'Instrumenting multi-agent organisations with organisational artifacts and agents: Giving the organisational power back to the agents', *Journal of Autonomous Agents and Multi-Agent Systems*, **20**(3), 369–400, (May 2010).

[18] Jomi F. Hübner, Jaime S. Sichman, and Olivier Boissier, 'Developing organised multi-agent systems using the MOISE+ model: Programming issues at the system and agent levels', *International Journal of Agent-Oriented Software Engineering*, **1**(3-4), 370–395, (2007).

[19] Hans Kelsen, *General Theory of Norms*, Oxford University Press, 1991.

[20] Hans Kelsen, *Pure Theory of Law*, The Law Book Exchange, New Jersey, 2009.

[21] Alessandro Ricci, Mirko Viroli, and Andrea Omicini, 'Programming MAS with artifacts', in *PROMAS @ AAMAS 2005 - Programming Multi-Agent Systems*, eds., Rafael P. Bordini, Mehdi Dastani, Jürgen Dix, and Amal El Fallah Seghrouchni, volume 3862 of *LNAI*, pp. 206–221. Springer, (2006).

[22] Yoav Shoham, 'Agent oriented programming', *Artificial Intelligence*, **60**(1), 51–92, (1993).

Towards a Distributed Data-Sharing Economy

Samuel R. Cauvin, Martin J. Kollingbaum, Derek Sleeman, and
Wamberto W. Vasconcelos

Dept. of Computing Science, University of Aberdeen, U.K.
{r01src15, m.j.kollingbaum, d.sleeman, w.w.vasconcelos}@abdn.ac.uk

Abstract. We propose access to data and knowledge to be controlled through fine-grained, user-specified explicitly represented policies. Fine-grained policies allow stakeholders to have a more precise level of control over who, when, and how their data is accessed. We propose a representation for policies and a mechanism to control data access within a fully distributed system, creating a secure environment for data sharing. Our proposal provides guarantees against standard attacks, and ensures data-security across the network. We present and justify the goals, requirements, and a reference architecture for our proposal. We illustrate through an intuitive example how our proposal supports a typical data-sharing transaction. We also perform an analysis of the various potential attacks against this system, and how they are countered. In addition to this, we provide details of a proof-of-concept prototype which we used to refine our mechanism.

1 Introduction

Large scale data sharing is important, especially now, with more open societies of components such as Smart Cities [25,3] and the Internet of Things [1,11] creating data sharing ecosystems. Currently, data access policies tend to be managed centrally, which comes with a number of problems such as information ownership and reliance on a centralised authority.

In [16] the author suggests taking a "data-oriented view" and developing methods for treating access policies and data items as a single unit. This allows data to prescribe their own policies, which can be checked when the data is shipped around between data management systems. Such a proposal of tying policies directly to data is described by, e.g., [24] as *policy-carrying data* that allows the specification of fine-grained policies for data items. In this paper, we present novel policy-based data sharing concepts for distributed peer-to-peer networks of data providers and consumers. Our working hypothesis is that it is possible to (a) create a fully distributed mechanism to facilitate data sharing with security guarantees, and (b) to implement a fine-grained control over how data may be exchanged between stakeholders.

We propose access to data and knowledge to be controlled through fine-grained, user-specified explicitly represented policies. These policies are used to regulate data exchange in a peer-to-peer environment in which some peers have data which they want to provide (called Providers) and some peers have data which they want to acquire (called Requestors). Providers set policies that establish how their data can be accessed and by whom. These policies can be defined with different levels of granularity, allowing peers precise control over their data.

Our policies may express general regulatory statements such as, for example, “no drug records and medical records can be obtained by the same party”, or more specific, such as “I will only provide 10 records to each person”. Fine-grained policies allow stakeholders to have a more precise level of control over who, when, and how their data is accessed. We propose a representation for policies and a mechanism to control data access within a fully distributed system, creating a secure environment for data sharing. We discuss data as if it were stored in a database, but this could be expanded to cover any form of structured information.

These policies will be enforced by a distributed infrastructure of “policy decision points” (taking inspiration from the traditionally centralized XACML PDP architecture [7]) throughout the network. We regard a data exchange or sharing activity between peers (provider and requestor) as a transaction. Transactions are recorded and are an important means for checking policy compliance. During a data request, transaction records are taken into account to test whether a requestor complies with the policies specific to such a request and the data involved. Due to the distributed nature of making policy decisions at peer-to-peer network nodes, a requirement for encrypting information components to be exchanged for this decision process arises. We take inspirations from encryption concepts in distributed applications, such as CryptDB [20], BlockChain [21,10] and Bitcoin [18].

This paper focuses on providing a simple case example demonstrating the feasibility of this mechanism, including reasoning on encrypted data using the mechanism. Ours is a starting point from where more sophisticated policy representations and reasoning mechanism can be developed, with more expressive and flexible representations for policies to provide greater control to the user. The work presented here is an initial investigation into this kind of reasoning process which can be made more sophisticated, to address arbitrary deontic reasoning and more complex interactions.

Section 2 details a general example of a simple transaction between two parties and then discusses the key components and concepts within our solution. Section 3 provides an overview of the requirements and architecture of the system. Section 4 describes the detail of a transaction scenario, discussing how each part of the mechanism is involved in the process. Section 5 evaluates the mechanism’s resistance to standard attacks. Section 6 discusses a proof-of-concept implementation of our solution. Section 7 provides an overview of related research. Section 8 discusses the limitations of our solution, provides overall conclusions, and outlines future work.

2 Policy Compliance

In our approach, so-called “transaction records” play an important role in whether any action related to sharing data is compliant with the policies relevant for this data. To illustrate how our mechanism performs a simple transaction, we consider a general case where two parties, a so-called “requestor” and a “provider”, want to exchange data. Such a transaction represents a secure, tamperproof interaction between requestor and provider. Following this example we discuss transaction records (Section 2.1), numerical encoding of data elements (Section 2.2), and policies (Section 2.3) in more detail.

The requestor will be represented by *R*, the provider by *P*, and the data element by *D*. The transaction will proceed as follows:

1. Requestor *R* sends a data request for data *D* to provider *P*.
2. *P* processes this request, and if the provider possesses *D*, it will create a list of policies relevant to *D* or *R*. If any policy in this list prohibits sending *D* to *R* (regardless of transaction records), then the data request will be denied, a transaction record will be generated and sent to *R*, and the process will terminate here. If not, *P* will send a message to *R* containing the policies associated with *D*.
3. *R* will reason on these policies to determine which transaction records are "relevant" (see Section 2.4). To achieve this, the mechanism loops through each policy and extracts a list of unique data elements referred to in the policy. At the end of this loop the list will contain each "relevant" data element (encoded as a number as discussed in Section 2.2).
4. The mechanism will then identify which of *R*'s transaction records are relevant using Algorithm 1 (in Section 2.4).
5. *P* receives records from *R* and needs to determine if any of the records prohibit the provision of *D*. For each of *P*'s policies the mechanism can process each record to determine if the data element is subsumed by a data element of *P*, and thus if the conditions of *P* hold. While processing, a cumulative total for each type of record will be kept. This total can be calculated without decrypting, as it requires only basic arithmetic on numerical entries. After processing all records, this total will be checked against the policy to determine if it holds or not. The order of policies is important, as earlier policies supersede later policies, that is as soon as a policy is triggered then the reasoning process can cease.
6. If this policy is a P (permit) policy, then sending *D* (the requested data and a record of the transaction, encrypted in a single package) to *R* is approved, and *D* will be sent from *P* to *R*.
7. *R* will decrypt the package, adding the transaction record to its records, and store the data. This single encrypted package is received by the mechanism, ensuring that the transaction record will be stored – as ignoring it will prevent receipt of data.

2.1 Transaction Records

Our policies relate data collections and events following the usual semantic of norms/policies (e.g., [19,22]), whereby events and their authors are explicitly represented (together with additional information such as time, location, duration, etc.) and used to check for (non-) compliance. In our proposal, events are named transaction records, and are stored encrypted within the information kept by each peer. Whenever a policy needs to be checked for its applicability, a subset of transaction records is retrieved from the encrypted storage, and used to compare the credentials/identification of the peer, assess the applicability to data elements currently available and verify if the conditions of our policies hold.

Transaction records are tuples of the form $\langle dataset, m \rangle$. The $dataset$ component refers to an ontological term, which is defined in one of the ontologies held by peers. Policies and transaction records refer to descriptions of data elements – these are labels

describing, for instance, fields of a data base or names of predicates of an ontology [4]. We adopt a numeric representation for these labels, and rather than using, for instance, *nameOfClient* or *fatherOf* (to represent, respectively a field of a database or a predicate), we use a numeric encoding.

2.2 Numerical Encoding of Data Elements

Policy checking is performed on encrypted transaction records without decrypting them, and performing operations on encrypted numerical data is far easier than on encrypted string data. To facilitate this, we introduce a numbering scheme that represents such a hierarchy of concepts and sub-concepts, including the encoding of concept properties. For this, we assign to each level in the subsumption hierarchy found in an ontology a code out of the range of [0 .. 99]: when we use the notation $[00..99]_1$, $[00..99]_2$, $[00..99]_3$, then we are expressing that a concept hierarchy has three levels (where the subscripts indicate levels), and each concept can relate to a maximum of 100 (0 to 99) sub-concepts. By concatenating the level codes from a top-level concept to a particular sub-concept, we arrive at a unique code for each concept in a hierarchy. Consider the taxonomy below (with the encoded number at the start of each line):

- 010000 Prescriptions
  - 010100 Name
  - 010200 Drugs
  - 010300 Patient Notes
    - 010301 Other Medications
    - 010302 Other Conditions
  - 010400 Renewal Date
- 020000 DrugX
  - 020100 Trial Number
  - 020200 Patient Notes
    - 020201 Other Medications
    - 020202 Other Conditions
  - 020300 Recorded Side-effects
  - 020400 Treatment Effectiveness
- 030000 Vehicles
  - 030100 Motorcycles
    - 030101 Owner

For example, in the above taxonomy, Vehicles is the third top level concept $[03]_1[00]_2[00]_3$. A concept below that, Motorcycles, is $[03]_1[01]_2[00]_3$ which indicates it is the first sub-concept of Vehicles. The size of each level and total number of levels can be increased, but this will also increase the size of each encoded number. The subsumption relation between two encoded numbers allows us to capture "is-a" relationships among concepts of a taxonomy, as in $030100 \sqsubseteq 030000$.

2.3 Policies

Policies enforce how data can be shared within the network. Some are network-wide (e.g., "no drug records and medical records can be obtained by the same party"), while

others can be specified by an individual provider (e.g., "I will only provide 10 records to each person"). These policies are stored by each peer locally.

We define our policies as follows:

Definition 1 (Policies). *A policy π is a tuple $\langle \mathbf{M}, \mathbf{I}, \mathbf{D}, \mathbf{P} \rangle$ where*

- $\mathbf{M} \in \{\mathsf{O}, \mathsf{F}, \mathsf{P}\}$ *is a deontic modality/operator, denoting an obligation (O), a prohibition (F) or a permission (P).*
- $\mathbf{I} \in \{id_1, \ldots, id_n\}$ *is a unique peer identifier*
- $\mathbf{D} \in \mathbf{T}$ *is a descriptor of a data element (cf. Def. 2)*
- $\mathbf{P} = L_1 \wedge \cdots \wedge L_m$ *is a conjunction of possibly negated literals (cf. Def. 3)*

A sample policy could be as follows: $\langle \mathsf{P}, id_1, 010000, 5, 0, 1 \rangle$. Our policies above refer to descriptions of data elements – these are labels describing, for instance, fields of a data base or names of predicates of an ontology [4]. We adopt a numeric representation for these labels, and rather than using, for instance, *nameOfClient* or *fatherOf* (to represent, respectively a field of a database or a predicate), we use a numeric encoding. Our taxonomies are subsets of natural numbers with a subsumption relationship, and are thus defined:

Definition 2 (Taxonomy). *A taxonomy $\mathbf{T} \subset \mathbb{N}$ is a subset of natural numbers. We define a reflexive and transitive subsumption relation $\sqsubseteq \subseteq \mathbf{T} \times \mathbf{T}$, over a taxonomy $\mathbf{T}$ to define its structure.*

An example of a taxonomy is given in Section 2.2.

Our policies allow the representation of *conditions* under which the policy should hold – this is what the component $\mathbf{P}$ of Def. 1 is meant for. We have designed a simple vocabulary of "built-in" tests which are relevant to our envisaged application scenarios, and these are defined below:

Definition 3 (Literals). *A literal L is one of the following, where $\mathbf{D} \in \mathbf{T}$ (a descriptor of a data element), $\circ \in \{<, >, \leq, \geq, =\}$ is a comparison operator, and $n \in \mathbb{N}$ is a natural number:*

- $noRec(\mathbf{D}) \circ n$ *– it holds if the number of retrieved instances of data element $\mathbf{D}$ satisfies the test "$\circ\, n$".*
- $lastReq(\mathbf{D}) \circ n$ *– it holds if the (time point of the) last retrieved instance of data element $\mathbf{D}$ satisfies the test "$\circ\, n$".*
- $lastAccess(\mathbf{D}) \circ n$ *– it holds if the (time point of the) last granted access to an instance of data element $\mathbf{D}$ satisfies the test "$\circ\, n$".*
- $\bot$ *and* $\top$ *represent, respectively, the vacuously false and true values.*

We make us of a simple account of time which can allow all events to be associated with a natural number.

In the remainder of our presentation, however, we make use of a "customised" version of policies, as these are more commonly used in our envisaged scenarios. We use the following shorthand:

$$\langle \mathbf{M}, \mathbf{I}, \mathbf{D}, (noRec(\mathbf{D}) < n \wedge noRec(\mathbf{D}') < n') \rangle \equiv \langle \mathbf{M}, \mathbf{I}, \mathbf{D}, n, \mathbf{D}', n' \rangle$$

Some examples of policies are as follows:

- $\pi_1 = \langle \mathsf{P}, id_1, 010000, 5, 0, 1 \rangle$, that is, peer id_1 is permitted to access 5 items of data element 010000; the remainder of the policy condition is idle (it imposes no further constraints).
- $\pi_2 = \langle \mathsf{P}, id_2, 020200, \infty, 0, 1 \rangle$, that is, peer id_2 is permitted to access unlimited (∞ stands for a very high natural number) items of data element 020200; the remainder of the policy condition is idle, that is, $noRec(0) < 1$ imposes no further restrictions.
- $\pi_3 = \langle \mathsf{P}, any, 010200, 5, 010000, 1 \rangle$ that is, any peer (denoted by the any identifier) is permitted to access 5 items of data element 010200; provided that they accessed less than 1 record of 010000.

This is a simple representation of policies which ignores the context of time and presents a simple example. The language of policies can be more expressive for the mechanism we are proposing. A more expressive language would allow more complex interactions between policies, which would also require a more complex reasoning process (we give potential expansions in Section 8).

In Def. 1 we discuss the notion of obligations, which can be thought of as deferred policies: actions to be taken (or not taken) after data has been received from a provider for a pre-specified period of time (or possibly indefinitely). For instance, an obligation could be defined that requires the requestor to provide 5 records of data element 010000 to the provider in exchange for 10 records of data element 020000. The encounter in Section 2 could also cater for situations where obligations can be transferred between parties. For example, with three parties A, B, and C: A provides data to B, and B is then obliged to provide data to A. B then provides data to C, and transfers their obligation to C. Now C is obliged to provide data to A, and B has no obligation to A.

We define in Equation 1 how to check if two data elements $\mathbf{D}$ and $\mathbf{D'}$ encoded in our numbering scheme of Section 2.2, are subsumed by one another. The definition makes use of two extra parameters, namely, BS which provides the size of each band (2, in the above example), and ZB which provides the number of zero-bands in $\mathbf{D'}$ (for instance, 020200 above has 1 zero-band $[02][02][00]$; calculating the number of zero bands is trivial for an unencrypted integer):

$$\mathbf{D} \sqsubseteq_{ZB}^{BS} \mathbf{D'} \text{ if, and only if, } \lfloor \mathbf{D}/10^{BS \times ZB} \rfloor = \lfloor \mathbf{D'}/10^{BS \times ZB} \rfloor \tag{1}$$

Each peer is provided a copy of the encoded ontology upon joining the network. If the ontology is too large, a subset could be provided containing concepts that the peer deals with and each transaction would provide the "vocabulary" of the requestor. In this way only a small amount of data is transferred when a new peer joins the network, but peers will slowly converge towards holding a complete ontology as transactions occur. Alternatively, the peer could be provided only with a URI; allowing them to download the full encoded ontology at any time.

Automatic encoding of the ontology is fairly trivial. The superclass-subclass relationships can be condensed into a simple tree structure; from this tree we can then count the maximum depth and maximum size at each depth to determine number of bands, and size of banding, respectively. This may take some time to complete, but this operation only has to be performed once on network initialisation.

Alternative numerical encoding mechanisms have been suggested that used ring theory, prime numbers, or multiples; however none seemed to precisely suit our needs.

Specifically, none could incorporate entailment information whilst retaining a mathematically simple comparison operation. Mechanisms of this type have been widely explored [6,13], and these mechanisms could replace the one currently proposed. For the purposes of our research we wanted to create a simple example encoding, however others could have been used.

2.4 Finding Relevant Transaction Records

The mechanism itself chooses relevant transaction records to send to a provider, the peer is unable to intervene. The challenge is ensuring that the records held by a given peer are tamper-proof; this is achieved by storing records in an encrypted format, using the numerical encoding in Section 2.2. Equation 1 allows identification of records that match a specific concept (or one of its parents). Using this information, and a reasoning process that references both policies and what is known about the requested data, a subset of relevant records can be identified and sent to a provider. On receipt of these records, the provider must also reason with them to determine if they violate any policies.

The mechanism identifies relevant records by looping through each transaction record and performing a numerical comparison operation, without decrypting the data. Each transaction has an associated data element, which is compared to each data element in the policies for the current transaction using Equation 1. If the test is passed, then the transaction will be retained as a relevant record. When all records have been processed, all relevant records will be sent to *P*. This process is detailed in Algorithm 1.

Algorithm 1 Finds Relevant Transaction Records

Require: Π (a set of policies), $Records$ (a set of records)
Ensure: $RelevantRecords$ (a set of relevant records)
procedure FINDRELEVANTRECORDS()
$RelevantRecords \leftarrow \emptyset$
for all $R \in Records$ **do**
for all $\pi \in \Pi$ **do** ▷ Each data type referred to in policies
if $encodedComparison(\pi, R)$ **then** ▷ $encodedComparison$ refers to Equation 1
$RelevantRecords \leftarrow RelevantRecords \cup \{R\}$
end if
end for
end for
end procedure

The mechanism must be able to detect potential violations and protect against them; either by updating policies, anonymising part of the data, or rejecting the request. When making this decision the mechanism will check if the users identity allows them to access the data, if they have fulfilled all past obligations, and if the records they have provided would prohibit them from receiving the requested data.

When deciding whether to share data, both ends of the transaction are black-boxed; this prevents either the requestor or provider from tampering with records. The en-

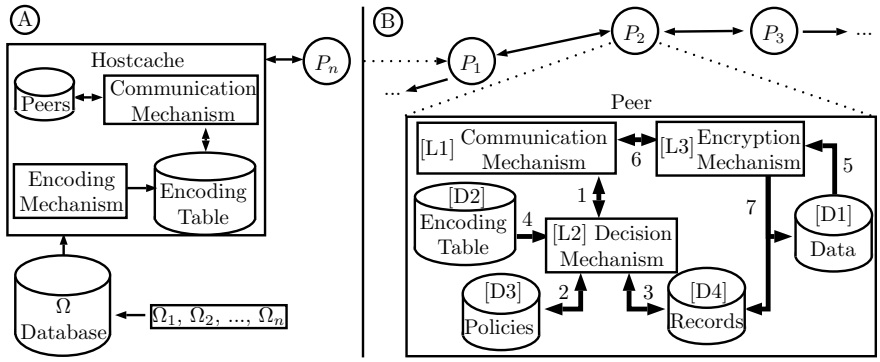


Fig. 1. Architecture

crypted records and (unencrypted) policies get passed into a black-box mechanism, which returns a boolean value to indicate if the transaction can go ahead. If the transaction is denied, then an encrypted record will be returned to the requestor that contains a justification (and full proof of reasoning) as to why it was denied. This can then "bootstrap" the reasoning process next time; as this record will be sent (by the requestor) as a relevant record. The provider can then examine the proof and decide if it still applies, reducing reasoning overheads.

The other challenge is designing the selection procedure in the mechanism so that just the right amount of information can be shared; since peer-to-peer connections are opportunistic, the less information sent the better – however enough has to be sent to allow the provider to make an informed decision about whether to share.

3 Requirements and Architecture

The hypotheses in Section 1 emphasise the aspects of the problem that we are concentrating on, and can be broken down further into the following requirements:

R1 To allow fine-grained (table and column level) control over data access policies.
R2 To ensure transaction records and data remain tamper-proof throughout the lifetime of a transaction.
R3 To allow operations to be performed on encrypted transaction records, without exposing those records to the user.
R4 To ensure that policies are enforced across the network and cannot be subverted to the advantage of an attacker.[1]

An architecture to meet these requirements is presented in Figure 1. The architecture above has two main components: the hostcache sub-architecture (A), and the peer sub-architecture (B).

[1] An attacker is any party (requestor, provider, or third party) who attempts to subvert the system.

It should be noted that our approach refers to data using database terms (tables, columns, and rows); however this is a specific case for our broader solution. While we assume that our mechanism will be used on data stored in a database, any knowledge base could be used instead.

The hostcache sub-architecture (A), which follows established peer-to-peer hostcache operations [2], has access to a collection of ontologies (obtained from many parties), which are input to the encoding mechanism. The encoding mechanism outputs the encoding table, which is a numerically encoded representation of the ontology (explained in Section 2.2). The hostcache also stores a collection of peer ids, each new peer that contacts the hostcache will have its peer id added to the collection. The hostcache processes requests from peers by generating ids, providing copies of the encoding table to peers and providing sets of potential neighbours to enquiring peers. The hostcache is a central element whose main functions are to generate ids for each peer, and to provide a list of potential neighbours on request. It also handles the one-time encoding of the underlying ontology.

The peer sub-architecture (B) is a collection of storage and logic components. The encoding table (D2) on the peer is obtained directly from the hostcache, and is only referred to by the decision mechanism (L2). The decision mechanism is responsible for performing the decision operations discussed later in this document (whether to provide data, what records are "relevant"). The peer also holds data (D1 – the data which it provides), records (D4 – encrypted transaction records), and policies (D3 – policies detailing how data is shared, discussed in Section 2.3). There is also the communication mechanism (L1) which handles message processing (both receiving and sending), generating data requests, and invoking the decision mechanism. Lastly is the encryption mechanism (L3), which can encrypt and decrypt data and record packages (but not records themselves) received from the network (discussed further in Section 4). While not noted in the architecture, each peer also holds an encrypted id, issued by the hostcache, that confirms who they are.

Each component in the peer sub-architecture is needed to fulfil at least one requirement. Requirement 1 needs the decision mechanism and policies. Requirement 2 needs all components *except* policies. Requirement 3 needs the decision mechanism, encryption mechanism, encoding table, and records. Requirement 4 needs the communication mechanism, encryption mechanism, and encoding table.

We engineer the behaviour of peers so as to make contact with the hostcache, establish neighbours, and then go into a loop responding to messages and requesting data. The protocol adopts non-blocking message exchanges, that is, peers do not wait for replies (as communication is unreliable and these may never arrive or be delivered). The interactions in sub-architecture B are numbered to represent a rough interaction protocol, but as interactions occur in a distributed environment they cannot be considered as sequential operations on a single peer. More accurately, there are four (main) paths through the architecture diagram for two interacting peers. Peer id_1, upon receiving a data request from Peer id_2, will follow steps 1, 2, 4, 1 (from the annotated arrows of Fig. 1). Peer id_2 will follow steps 1, 2, 3, 1. Peer id_1 will follow steps 1, 2, 4 and then 5, 6. Peer id_2 will then follow steps 6, 7.

4 Illustrative Scenario

We illustrate our solution with a scenario in which we consider two parties: *P* (the provider) and *R* (the requestor). The provider is a research lab that developed DrugX, and tracks prescriptions of DrugX. The requestor is a health authority who regulate all prescriptions for the region they operate in attempting to counteract the side effects of Drug X. This example uses a subset of the encoding table from Section 2.2.

The requestor wishes to get information on the trials carried out on DrugX by the provider, so sends a data request for ten 020000 (DrugX and subclasses) records. The provider checks its policies and finds nothing prohibiting the requestor's access to 020000, so the provider then sends the following (relevant) policies to the requestor:

- $\langle P, any, 010300, \infty, 020200, 1\rangle$ – Deny 010300 to anyone who has 020200
- $\langle P, any, 020200, \infty, 010300, 1\rangle$ – Deny 020200 to anyone who has 010300

The requestor loops through these policies and extracts the following data elements: 010300 and 020200. The requestor then has to check through their transaction records (the format is $\langle dataset, numberOfRecords\rangle$): $\langle 010100, 50\rangle$, $\langle 010301, 50\rangle$, $\langle 010302, 50\rangle$, $\langle 010100, 10\rangle$, $\langle 010200, 10\rangle$, $\langle 010400, 10\rangle$

Each relevant data element is then compared with the records to determine its entailment, following Equation1, that is, $010301 \sqsubseteq^{BS}_{ZB} 010300$, and $010302 \sqsubseteq^{BS}_{ZB} 010300$ hold; none of the remaining cases hold.

For each pair $(\mathbf{D}, \mathbf{D}')$ we must test both $\mathbf{D} \sqsubseteq^{BS}_{ZB} \mathbf{D}'$ and $\mathbf{D}' \sqsubseteq^{BS}_{ZB} \mathbf{D}$, as the test will only capture if the first element is a subclass of the second. Applying both tests allows both relationships to be captured. Of the six records two of them are found to be relevant: $\langle 010301, 50\rangle$ and $\langle 010302, 50\rangle$. These records are now sent to the provider to be reasoned on to determine if they violate any of their policies. This process is similar to the process performed by the requestor, so we will not discuss it in as much detail. Performing the same basic loop the mechanism determines that Policy 2 (Deny 020200 to anyone who has 010300) holds for both records. At this point, the provider can do one of two things: the Data Request can be rejected (a justification record will be generated and sent to the requestor), or part of the requested data can be omitted. The latter will be used in this situation, as the policy only prevents a specific part (020200) of the requested data (020000) from being sent.

The provider then generates records for the current transaction ($\langle 020100, 10\rangle$, $\langle 020300, 10\rangle$, $\langle 020400, 10\rangle$), and assembles the result package (containing 10 records of 020100, 020300, and 020400). These are then encrypted together using the requestor's public key[2] and sent to the requestor. The requestor's mechanism receives this package and decrypts it using the requestor's private key. The "receipt" is added to the requestor's collection of transaction records and the mechanism returns the extracted data to the requestor, completing the transaction.

[2] This is an extra security precaution, assuming that all peers have Public/Private Key pairs ensures that data can be sent across a peer-to-peer network securely.

5 Analysis of our Solution

We evaluated our proposal by exploring many cases and concluded that there was no incentive for any of the participants to subvert the system, as it provided no advantages. Below we provide an analysis of our proposal against classic attacks.

- *Impersonation* – All peers, in order to join the network, must be given a unique encrypted id by the host cache. Ids cannot be falsified as only the hostcache has keys to generate these appropriately and the chances of falsifying ids coherently are very low.
- *Modification of policies* – Providers could modify policies during transactions, however doing so could cause them to receive irrelevant transaction records. These irrelevant records could cause them to make an incorrect decision to provide or withhold data, which they would have no incentive to do.
- *Modification of transaction records* – Transaction records cannot be tampered with as they are encrypted throughout exchanges; attempts to tamper with records would need to break the encryption mechanisms.
- *Man in the Middle* – Transaction records and data both travel encrypted. Policies are transmitted unencrypted, but it would be trivial to create a RSA-like encryption to transmit them. Man-in-the-middle can not access the data as it travels encrypted.
- *Denial of Service (DOS)* – Requiring a hostcache creates a vulnerability to DOS attacks, however this DOS would only affect new peers joining the network. Existing peers in the network would be able to function as normal. A DOS could also target individual peers, but this will not have a major effect on the rest of the network.
- *Subvert timestamp in records (Provider)* – This timestamp is generated by the mechanism, so cannot be altered. The provider could potentially alter it by garbling the record, but this would only serve to disadvantage them in the future.
- *Provider sends malformed record* – A malformed record will never be considered a "relevant" record, as it cannot be processed properly by the mechanism, so if they are sent, they will be ignored. There is no incentive for a peer to send malformed records. To prevent this record from remaining indefinitely a record purging functionality periodically scans the set of records and discards those elements which cannot be processed/parsed.
- *Requestor does not record transaction* – The mechanism forces transaction records to be stored. Providing the data and updating the set of records are two stages of an atomic operation carried out within the black-box mechanism.
- *Code tampering* – Tampering with code is impossible, as it is provided as a black-box.
- *Record fabrication* – Records could be fabricated, but the chances of producing anything meaningful are very low, since these have to be encrypted and the peers do not hold the keys or indeed have access to the encryption mechanism by itself.
- *Sybil Attack*[3] */Fake peer generation* – The only purpose to generating extra peers would be to generate fake records for yourself, but there is no benefit from having

[3] A sybil attack [8] happens when one of the participants generates many fake ids to skew the balance of power in one's own favour, as in, for instance, voting.

extra records as these will not make approval more likely, moreover, it could cause data requests to be rejected.

– *Data Modification* – After a peer receives data from another peer, that data is no longer under the control of the data provider. We propose a way of mitigating this by adding the concept of data "ownership". All data within the network can be stored in "packages" encrypted with the id of the original owner. Anyone can decrypt these packages to get the data, but they are only able to encrypt data packages with their own id. This means that the original source of the data is in no way associated with the data after it has been modified by a third party.

Our solution incorporates a small amount of centralisation: a one-time check-in when connecting to the network, to aid with system functions. It may be possible to design a system where this is not the case, but we would have to make trade-offs (no verified identities, no shared encoding, cold-start issues, etc.) to achieve this. This minor centralisation ensures that no one "owns" all the data within the system, and also creates a robust network for data exchange; the only contact with a central authority (hostcache) is when a peer joins the network, after that no data is sent to the hostcache.

6 Proof of Concept Implementation

We investigated the design space by creating a proof-of-concept prototype[4] to perform the operations of single peer with a set of simulated neighbours. Our prototype does not implement full message passing, but does demonstrate the mechanism which we have described. The prototype is implemented in Java, so some definitions have been adapted to fit with object-oriented programming concepts. The cryptography implemented in the prototype is not a full encryption mechanism, but simulates one through the use of numerical objects that can have simple mathematical operations performed on them without exposing their value. If we ignore these adaptations, our implementation follows the peer architecture (part B of Fig. 1) faithfully.

To reflect the modularity of our architecture we have introduced features to customise the simulation using a number of parameters, currently specified as variables within the code. These parameters supply the (ontology) encoding table, data, records, and policies of each neighbouring peer. The simulation itself tracks a number of metrics to provide an analysis of performance. The policies implemented within our prototype follow our policy language provided in Def. 1, specifically they make use of the shorthand we describe in Section 2.3.

We have also performed a feasibility analysis by using this prototype to simulate an extended version of the scenario from Section 4. The scenario considers a single peer attempting to get data from four neighbours that each have data and policies. This simulation completes in a single cycle (each neighbour is queried for each desired data item once) with all of the requested data being received. The prototype tracks which peers provided the data, allowing this to be compared to their policies; through this we observed that policies were not violated at any point.

[4] The source code for our implementation is `https://github.com/Glenugie/REND-Peer`

Using our prototype we tracked total number of messages sent between peers, total simulation time, and the minimum, average, and maximum size of messages. Message sizes are given in quantity of numbers transferred, with encrypted numbers taking twice the space, and each array adding an extra number as overhead. 40 messages were exchanged, with a minimum size of 1 (initial data request), an average size of 4.5, and a maximum size of 17. The simulation took a total of 10 milliseconds to complete, 2 milliseconds of which was the single cycle; the other 8 were network initialisation. This time could be considered inaccurate as we envisage our mechanism running on a large number of devices with little computing power; rather than the one powerful device that our simulation was run on.

Implementing this prototype allowed us to locate and correct a number of inconsistencies in our mechanism. One such correction was to apply the encoded number comparison from Equation 1 in pairs to capture entailment in both directions.

7 Related Work

Our investigation taps onto many disparate areas such as Smart Cities [25,3], Internet of Things [1,11], BlockChain [21,10], bitcoin [18], and encryption [9,20]. Below we review the work that we consider most relevant.

This paper draws upon the techniques and methods reported in [19], but it has a significantly different focus, and most importantly, provides a distributed solution which will scale up and is resilient to many kinds of attacks. Our proposal extends the idea of combining data and access policies into one single computational entity with benefits such as increased control over how your data is used. There have been other research threads which also use the term "Policy Carrying Data" [23,24], which suggests similar concepts but without the focus on a distributed environment. They instead focus on maintaining data policies when the data, and policies, are uploaded to the cloud.

Berners-Lee makes a case for an online Magna Carta [15] to protect the openness and neutrality of the internet. The work being proposed here attempts to develop a mechanism to support the normative principles promoted in Berners-Lee's design [17].

Role Based Access Control (RBAC) could be seen as a similar line of work, though with a stronger focus on a controlled environment. While work has been pursued to begin addressing RBAC in a distributed environment [5,12,14,22], it has not been completely resolved.

One candidate for operations on encrypted data is homomorphic encryption schemes [9] which are applicable to our proposal. This method of encryption allows operations to be applied to encrypted data without decrypting. One limitation of this approach is that a data request must specify the amount of data to be retrieved, and the result will either be truncated or padded out. This method is semantically secure, i.e. given a cipher c that encrypts either m_0 or m_1, adversary α (when given the two choices) has probability of $\frac{1}{2} + \epsilon$ of guessing correctly. ϵ, called α's advantages should be negligible, else (informally) α has "broken" the semantic security.

Another candidate is CryptDB [20], though this is less suited to the required context. CryptDB relies on a trusted proxy to process a user's query and send it to the database management system (DBMS), which then returns the decrypted result. This

seems problematic, as the proxy returns the result in a decrypted format (so, while the DBMS has not seen anything decrypted, the decrypted result could be intercepted between proxy and user).

We note a substantial overlap between our proposal and initiatives such as BlockChain[5] and Bitcoin[6]. BlockChain is a permissionless distributed database [21,10] based on the bitcoin protocol [18] that achieves tamper resistance by timestamping a hash of "batches" of recent valid transactions into "blocks". Each block references the prior timestamp, creating a cryptographically enforced chain. Blockchain requires either a group of always-on data-store nodes, or for every individual "peer" to store a copy of the full chain. There are important similarities between BlockChain and our proposal, but BlockChain is centralised in nature and has high storage requirements on data store nodes.

8 Conclusions, Discussions, and Future Work

We proposed a solution to enable the control of data through fine-grained, user-specified access policies. This solution was designed to operate in a peer-to-peer scenario in which some peers have data which they want to provide (called Providers) and some peers have data which they want to acquire (called Requestors). Providers can set "policies", i.e. rules which govern how their data can be accessed and also by whom. These policies will be enforced by mechanisms throughout the network.

For simplicity we assume that a fixed ontology is provided on network initialisation, and that this is then encoded by the hostcache. In the future, it would be possible for this to be extended to a dynamic ontology where each peer reports their sub-ontology on joining the network, which is then added to the master encoding table. This fixed ontology allows for the correct banding size and depth to be determined for the encoding. If the encoding is dynamic, one shortcoming is that it is then possible to run out of encoding space; this can be offset by choosing a high starting size but this will increase message size.

The mechanism and example that we have presented in this paper consider a simple case with a number of limitations which can be improved upon through a number of extensions, some of which we have sketched. Below we present some potential extensions of the proposed mechanism.

Our mechanism currently only allows a requestor to (implicitly) accept or reject policies within a transaction; if they reject the policies specified by the provider they simply do not send relevant records to the provider. In the future we could implement a "policy negotiation" phase, in which requestor and provider can propose and counter-propose policies to attempt to reach an agreement. For instance, the provider could propose an obligation which requires the requestor to provider 10 temperature readings. The requestor could counter-propose that they only provide 5 temperature readings. This process can continue until an agreement is reached, or either party withdraws.

We have considered the notion of obligations, which can be thought of as deferred policies: they are actions to be taken (or not taken) after data has been received from a

[5] `https://blockchain.info/`
[6] `https://bitcoin.org/en/`

provider for a pre-specified period of time (or possibly indefinitely). Obligations could also be set to expire when certain conditions are satisfied (not just time-related), for instance once an obligation has been triggered a certain number of times. We could consider a more complex system where multiple obligations can be attached to a single piece of data, and each obligation can be individually negotiated. Another possibility would be to allow obligations to be assigned to the provider (and not just the requestor). This would allow obligations such as "If I send data to you, then I am obliged to keep you updated if that data changes." This could either be proposed by the provider or requestor during negotiations.

Another extension is automated record purging and clean-up. Peers want to hold a minimal set of records (as they take up storage space), so there needs to be an operation to purge records that are no longer useful. Each peer would purge its own records periodically. Records with unfulfilled obligations will always be kept (another incentive to fulfil obligations, as otherwise your storage space will get filled quickly). Peers could also perform record compaction, merging equivalent records (for example, $\langle 010200, 20\rangle$ and $\langle 010200, 30\rangle$ become $\langle 010200, 50\rangle$).

Our implementation currently only addresses one peer, but we have already started looking into ways of simulating realistic P2P networks, using a technology such as PeerSim[7], which allows hundreds of thousands of peers to be simulated efficiently.

Our policy language is a starting point, and we have many possible extensions we would like to explore to provide finer-grained control but with adequate computational (performance) features. We have considered extensions that allow policies to have a time component. We also plan to provide reasoning mechanisms that allow users to see what the consequences of accessing a given piece of data are. As part of this we would also need to introduce more complex reasoning into the mechanism, allowing it to deal with complex interactions between policies. The mechanism could also be extended to allow policies to target groups of users; the present formalisation considers each peer to be an independent agent. Extensions of the reasoning mechanism could also allow us to provide tools for peers lacking technical knowledge to construct policies to suit their needs.

References

1. Atzori, L., Iera, A., Morabito, G.: The internet of things: A survey. Computer networks 54(15), 2787–2805 (2010)
2. Buford, J., Yu, H., Lua, E.K.: P2P networking and applications. Morgan Kaufmann (2009)
3. Caragliu, A., Bo, C., Nijkamp, P.: Smart cities in europe. Journal of Urban Technology 18(2), 6582 (2011)
4. Chandrasekaran, B., Josephson, J.R., Benjamins, V.R.: What are ontologies, and why do we need them? IEEE Intelligent systems (1), 20–26 (1999)
5. Cheng, Y., Park, J., Sandhu, R.: A user-to-user relationship-based access control model for online social networks. In: Data and applications security and privacy XXVI, pp. 8–24. Springer (2012)

[7] `http://peersim.sourceforge.net/`

6. Curé, O., Naacke, H., Randriamalala, T., Amann, B.: Litemat: a scalable, cost-efficient inference encoding scheme for large rdf graphs. In: Big Data (Big Data), 2015 IEEE International Conference on. pp. 1823–1830. IEEE (2015)
7. Dhankhar, V., Kaushik, S., Wijesekera, D.: Securing Workflows with XACML, RDF and BPEL. In: Proceedings of the 22Nd Annual IFIP WG 11.3 Working Conference on Data and Applications Security. pp. 330–345. Springer-Verlag, Berlin, Heidelberg (2008), http://dx.doi.org/10.1007/978-3-540-70567-3_25
8. Douceur, J.R.: The sybil attack. In: Peer-to-peer Systems, pp. 251–260. Springer (2002)
9. Gentry, C.: Computing arbitrary functions of encrypted data. Communications of the ACM 53(3), 97–105 (2010)
10. Grigorik, I.: Minimum viable block chain. "https://www.igvita.com/2014/05/05/minimum-viable-block-chain/" (Accessed On: 2014)
11. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems 29(7), 1645–1660 (2013)
12. Hansen, M.: Top 10 mistakes in system design from a privacy perspective and privacy protection goals. In: Privacy and Identity Management for Life, pp. 14–31. Springer (2011)
13. Harrison, J.: Theorem proving with the real numbers (1996)
14. Karjoth, G., Schunter, M., Waidner, M.: Platform for enterprise privacy practices: Privacy-enabled management of customer data. In: Privacy Enhancing Technologies. pp. 69–84. Springer (2002)
15. Kiss, J.: An online magna carta: Berners-Lee calls for bill of rights for web. The Guardian 12 (2014)
16. Landwehr, C.: Privacy research directions. Communications of the ACM 59(2), 29–31 (2016)
17. Lee, B.T., Fischetti, M.: Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor. Harper San Francisco (1999)
18. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. www.cryptovest.co.uk (2008)
19. Padget, J., Vasconcelos, W.W.: Policy-carrying data: A step towards transparent data sharing. Procedia Computer Science 52, 59–66 (2015)
20. Popa, R.A., Redfield, C., Zeldovich, N., Balakrishnan, H.: CryptDB: Processing queries on an encrypted database. Communications of the ACM 55(9), 103–111 (2012)
21. Postscapes: Blockchains and the internet of things. http://postscapes.com/blockchains-and-the-internet-of-things (Accessed: March, 2016)
22. Sackmann, S., Kahmer, M.: ExPDT: A policy-based approach for automating compliance. Wirtschaftsinformatik 50(5), 366 (2008)
23. Saroiu, S., Wolman, A., Agarwal, S.: Policy-carrying data: A privacy abstraction for attaching terms of service to mobile data. In: Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications. pp. 129–134. ACM (2015)
24. Wang, X., Yong, Q., Dai, Y., Ren, J., Hang, Z.: Protecting Outsourced Data Privacy with Lifelong Policy Carrying. In: 10th IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing, HPCC/EUC 2013, Zhangjiajie, China, November 13-15, 2013. pp. 896–905 (2013), http://dx.doi.org/10.1109/HPCC.and.EUC.2013.128
25. Zheng, Y., Capra, L., Wolfson, O., Yang, H.: Urban computing: concepts, methodologies, and applications. ACM Transactions on Intelligent Systems and Technology (2014), http://dl.acm.org/citation.cfm?id=2629592

Modelling patient-centric Healthcare using Socially Intelligent Systems: the AVICENA experience

Ignasi Gómez-Sebastià[1], Frank Dignum[2], Javier Vázquez-Salceda[1], and Ulises Cortés[1]

[1] Department of Computer Science. Universitat Politècnica de Catalunya (BarcelonaTech), Spain
igomez,jvazquez,ia@cs.upc.edu
[2] Department of Information and Computing Science, Universiteit Utrecht, Netherlands
F.P.M.Dignum@uu.nl

Abstract. One of the effects of population aging is the increase in the proportion of long-term chronic diseases, which require new therapeutical models that mostly take place at the patients' home rather than inside a health care institution. This requires that patients autonomously follow their prescribed treatment, which can be especially difficult for patients suffering some kind of cognitive impairment. Information technologies show potential for supporting medication adherence but the main challenge is the distributed and highly regulated nature of this scenario, where there are several tasks involving the coordinated action of a range of actors. In this paper we propose to use socially intelligent systems to tackle this challenge. These systems exhibit, understand, and reason about social behaviour, in order to support people in their daily lives. Such systems present an opportunity when applied to information technologies for supporting treatment adherence. We explore how concepts of socially intelligent systems, including social practices and social identities, can be applied to AVICENA, a ongoing project to create a platform for assisting patients in several daily tasks related to their healthcare. We first introduce AVICENA, briefly describe our previous attempts to model the system from an organizational perspective and an institutional one and discuss some of the limitations found in those models. Then the core concepts of socially intelligent systems are introduced and we show how they can be applied to create a socially aware framework for supporting medication adherence.

Keywords: Multi Agent Systems, Social Intelligence, Assisted Living

1 Introduction

One of the main challenges that national healthcare programs will face in the near future is population ageing (i.e., the increase of the proportion of old people

within the total population). In the European Union the size of the population aged between 65 and 80+ years at this moment is 80 million, but studies indicate that this number may double by 2050 [28]. In the United States of America the group of older people (aged 60+ years) is estimated to grow from the current 11% to a 22% by 2050 [24]. Moreover this is not just a problem in developed countries, as population ageing is also present in developing countries and might have an even bigger impact in those countries.

One of the impacts of population ageing is the epidemiological shift in disease burden, from acute (short-term, episodic) to chronic (long-term) diseases. From the patients' perspective, chronic diseases imply lenghty treatments often involving the combination of various medications to be taken at different times. It is undeniable that many patients experience difficulties in following treatment recommendations, and poor adherence to these long-term therapies compromises their effectiveness and may even become a cause of death. Adherence to long-term therapy for chronic illnesses in developed nations averages 50%. In developing countries, the rates are even lower [31]. Adherence rates are typically higher in patients with acute conditions, as compared to those with chronic conditions, with adherence dropping most dramatically after the first six months of therapy and in prophylaxis [22]. There are many reasons why patients do not follow their therapy as prescribed. One of the reasons is that they cannot tolerate the (long-term) side effects such as loss of hair or constant feeling of tiredness. It may also be that the high cost of some medicines prohibits acquisition of their medication. Where a condition is asymptomatic (such as hypertension), the patient may be lulled into thinking that their treatment has worked and that they no longer require to take their medication or follow their diet; distracted by the hectic pace of everyday life, perhaps they simply forget to take their pills.

From the national healthcare programs' perspective, the epidemiological increase of chronic diseases implies the need of a major shift of the programs, from the current one centered on rapid response to episodic, acute ilnesses where most of therapies and treatments are managed and delivered inside the official institutional care setting, into one where most of the medical therapies for managing chronic diseases (*e.g.*, hypertension, diabetes, depression, Parkinson, *etc.*) are performed away from the institutional care setting, typically at home. This distributed approach to daily care requires patients, especially elderly, to be capable and committed to autonomously taking various medications at different time intervals over extended periods of time. This can easily lead to forgetfulness or confusion when following the prescribed treatment, especially when the patient is suffering multiple pathologies that require a treatment with a cocktail of drugs. This gets worse when elderly suffer a cognitive impairment. Both concordance and adherence management are of high priority, having a significant effect on the cost effectiveness of therapy. This is especially important where there are disorders with high healthcare costs, such as oncological diseases, psychiatric disorders, HIV, geriatric disorders or dementia. Initiatives attempting to address medicine non-adherence promote patient involvement in treatment decisions but

remain ineffective with older patients or with patients with cognitive disorders. Interventions using applied high-technology show potential for supporting medication adherence in patients with diseases that require poly-pharmacological treatment, as they could help to reach optimal cooperation between patients and the healthcare professionals.

In this context, Assitive Technologies (AT) have been able to provide successful solutions on the support of daily healthcare for elderly people, mainly focused on the interaction between the patient and the electronic devices. However, the distributed approach that such kind of healthcare has to follow in the current socio-economical setting requires more complex AT designs that go further than the interaction with a tool and are able to focus on the relationship between the patient and his social environment: caretakers, relatives, health professionals. In this paper we describe how AVICENA, a patient-centric AT system to support patients in their daily healthcare, may be enhanced into a socially aware system that promotes treatment adherence by keeping track of the patient's motivations. Next section describes AVICENA. Then in §3 we introduce the core concepts of socially intelligent systems that we will use for our solution. §4 shows how these concepts are used to convert AVICENA into a socially-aware system to support medication adherence. In §5 we discuss some related work and we end with some final conclusions and future work.

2 AVICENA

AVICENA is an ongoing project that proposes the development of an innovative m-Health [17] platform and well-tailored personalized services to substantially improve chronic patients' medication and treatment adherence. *AVICENA* offers the opportunity to solve the patient's non-adherence to treatments by encouraging self-management of the treatment and promoting the continuity of therapeutic regimen, reducing costs to the patient, the caregivers and the health system. *AVICENA* focuses on developing innovative control mechanisms for collaborative, adaptive, dynamic and user centred medical concordance assessment and management systems at preferred environments and highly cooperative, intuitive patient/machine/pharmacist/doctor interfaces over a network. The *AVICENA* platform (depicted in figure 1) includes:

- **a Smart pill dispenser** that provides the medication at the prescribed times. It controls missed doses via integrated sensors, controls the drug stock and contains a reasoning engine offering Smart services,
- ***AVICENA* mobile app**, empowering users with the ability to self manage their treatment, obtaining tailored information and feedback depending on their medical treatment adherence,
- **a new care model** involving all the stakeholders in the chronic treatment process and in the assessment and management of the treatment adherence,
- ***AVICENA* social network** connects all the stakeholders in the care process (i. e., patients, clinicians, caregivers and pharmacists).

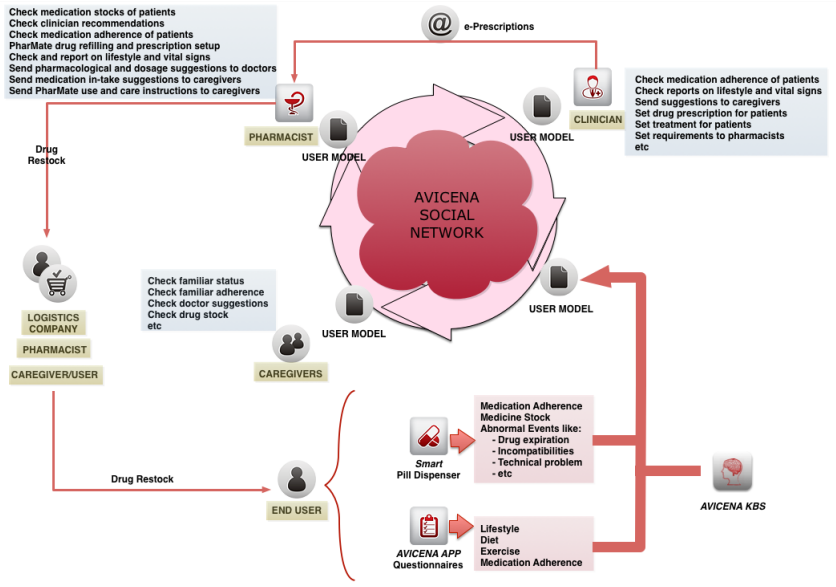


Fig. 1. *AVICENA* Architecture

The main goal of *AVICENA* is to improve individuals' adherence to medical treatments. A major application of the system will be the assistance of elderly individuals with chronic systemic diseases for which complex drug therapies are prescribed. In fact, several factors may affect adherence to medical treatments of this individuals, among which memory failures and psychological frailty play a relevant role. Indeed, cognitive disorders and psychopathological alterations such as mood fluctuations, anxiety and reduced efficiency of control mechanisms, are relatively frequent in this clinical population. *AVICENA* should directly influence the caregiver-patient efficiency to follow medical prescriptions by improving both the communication with the other agents of drug therapy assistance (e.g., physician, pharmacist) and the capacity of the caregiver-patient system to recognize and cope with factors likely related to reduced compliance.

In previous work [14] we presented an early version of AVICENA's model based on the ALIVE [3] framework. In that first stage of the work we focused on the organizational model, and the ALIVE framework eased the design of the social network built around the patient (i. e., patient, doctor, health insurance company, pharmaceutic, delivery person, domotic house, intelligent medical dispenser and medical monitor) through a rich organisational, role-based model based on OperA [10]. All scenarios roles were clearly defined, including their responsibilities and dependencies. But the normative model was still a simple one, and it was properly extended in [13]. Figure 2 shows some sample norms. The expected behavioural patterns to be abided by the actors in the scenario

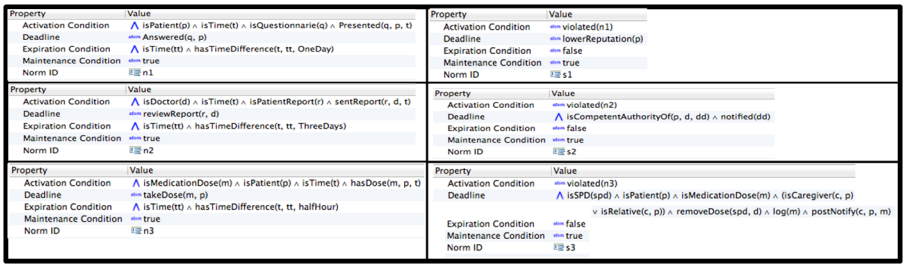


Fig. 2. Example of norms in *AVICENA* (source: [13]).

(including both human actors and computational agents) were properly connected to both constitutive and regulative norms, and an institutional monitor was set up to be able to infer the institutional state of an AVICENA setup. As a result we had a rich model which described the system from both a functional, organizationally-oriented perspective, an an institutional perspective. Expected behaviour for al actors was clearly stated, and for those cases of non-compliance, violation-handling norms were added. But the patient being obliged to follow her treatment does not lead to its compliance, and there is no effective sanction mechanism that can be placed in the scenario that can handle forgetful patients or unmotivated ones. Furthermore in the case of informal caregivers, there is no contract establishing their precise roles and responsibilities, and very often they play a key role in the daily treatment process, exceeding their responsabilities as relatives by partially or completely taking a caregiver role. Modelling these informal interactions is the main motivation of the rest of this paper.

3 Socially Intelligent Systems

The goal of the actors in the AVICENA scenario is for the patient to follow the treatment as accurately as possible while maintaining as much autonomy as possible. The second part of the goal is more interesting, because it leads to important social requirements. If the patient should be as autonomous as possible then her course of action should be driven mainly by internal motivations and not by contracts, obligations and prohibitions. Ideally we would like the patient to have an internal motivation and capabilities to follow the necessary treatment with the support of caregivers whenever needed. In order to get to this situation we need models that go beyond the functional goals of following the treatment and that also take into account social aspects of the actors. In particular we need the motives (achievement, affiliation, power and avoidance), values (leading to preferences for types of situations), social relations (power, trust, status, responsibility, etc.), social identity (image that one wants to give, leading to coherent behavior around values and practices, norms and roles) and social practices (indicating standard packages of social and functional behavior

combinations and interpretations of interactions that lead to both functional as well as social goals). We will motivate the use of all these aspects in the scenario and discuss some of their background and use in the scenario.

3.1 motives

As we already indicated above the goal of AVICENA is not just that the patient gets her treatment, which could be achieved by having a person or system take care of reminding the patient or even forcing the patient to follow the treatment. However, the autonomy of the patient requires the careful consideration of social aspects that surround the treatment. In [9] we argued that agents can only become truly social when we take into consideration all basic types of motives as defined by McLelland [21]. Besides the achievement motive, which can be thought to drive the traditional functional goals achievement (i.e. trying to achieve a state of the world) he distinguished the affiliation, power and avoidance motives. The affiliation motive underlies the need of people for (positive) social contact. This motive can be used (or abused) when a patient is not very mobile and is dependent on other people to come by for most social contacts. In that case a professional caregiver or family member that comes by to ensure that the patient follows the treatment (takes a pill or performs an exercise) also can fulfill the affiliation need of the patient as long as the person shows enough personal interest in the patient. The power motive is NOT about gaining social power over other people. It is actually meant to designate the drive people have to master capabilities and thus processes. E.g. sportsmen practicing skills and enjoying doing so comes from this motive. This motive can lead to the will to autonomously perform some actions related to a treatment. E.g. performing exercises that need physical or mental skills. The avoidance motive drives people to avoid unwanted situations. This plays a role in treatments when medicines might have negative side-effects or it is unknown how they will affect a patient. This uncertainty might lead a patient to avoid taking the medicines.

3.2 social identity

The second important aspect that needs to be taken into account is the social identity of a person. In short, the social identity of a person determines what other people expect from someone in certain contexts. The social identity consists of three elements: the perceived physical appearance, the identification with a stereotype and membership of social groups. The first element relates to what a person believes are his capabilities and thus what he believes other people expect him to do. I.e. if you are old you don't have to stand up for other people in public transport. If you consider yourself athletic you will take initiative when physical tasks have to be done for a group. If you consider yourself to be handicapped or ill (e.g. with heart failure) you might avoid going up stairs or taking a walk. The second element of a social identity indicates an ideal image (or prototype) that one strives to mirror. Thus one compares himself with the expected behavior of the ideal identity and also uses the expected behavior to guide one' s own

behavior. Thus if one believes that an ideal husband takes care of all broken appliances in the family home then the man will try to fix all of them or try to learn how to do this. He will consider himself bad if he fails in such tasks (even if they are not realistic). So, if a patient sees himself as a basically healthy person and healthy persons do not need assistance with any daily activity, the patient might refuse the support (even though he "knows" that he needs the support for the activity). This second element can be modeled with two parts; the first is the set of values that a person attaches to the ideal and that he therefore tries to uphold and the second is a set of social practices that he considers to be appropriate given this ideal. The social practices come again with their own set of norms and default behaviors and roles. We will discuss the social practices later in more detail. The third element of the social identity of a person is his group membership. If a person is part of a social group he will adopt the social practices of this group and uphold its values. In how far he does this depends on his role in this group. The captain of a basketball team is more likely to follow the social practices of the team than a substitute. Membership and status of a group can in themselves also be goals of a person. Thus being a good family member can entice a patient to accept advice of another family member.

3.3 social practices

The final aspect of social agents that we will include in our models is that of social practices. In our every-day life most of our behavior is governed by social practices. They are a kind of standardized way in which we conduct all kinds of interactions. They combine standard physical behaviors with standard social interpretations of this behavior. E.g. greeting a person in The Netherlands at work with a handshake shows respect and an understanding that the meeting is formal. Someone that you see every day or who you consider to be a peer/friend you will greet by just saying "Hi". Thus there is both a standard physical action as well as standard social meaning attached to a social practice. The fact that these are combined makes them convenient in a complex world as it avoids to have to reason about both physical and social aspects separately. The reason that they work is exactly because they are standard. Thus their usefulness derives from their use rather than some intrinsic value of the actions themselves. The existing theory on social practices is rather sparse (but see [29,25] for some background) and not geared towards the use of them in operational contexts. However we use this social science theory as starting point. They have proposed a representation of social practices based on three broad categories [16]: materials, meanings and competences.

- Material: covers all physical aspects of the performance of a practice, including the human body (relates to physical aspects of a situation).
- Meaning: refers to the issues which are considered to be relevant with respect to that material, i.e. understandings, beliefs and emotions (relates to social aspects of a situation)
- Competence: refers to skills and knowledge which are required to perform the practice (relates to the notion of deliberation about a situation).

Based on these ideas, we developed a model to represent social practices that can be used in social deliberation by intelligent systems. Obviously, as is the case with e.g. the representation and use of norms, other representations of social practices are possible given the many dimensions of the use of social practices. Our proposal, depicted in Figure 3, is especially suitable for use in agent reasoning. The components of this representation model are as follows:

Concrete Social Practice	**Family visit of youngest daughter**
Physical Context	
Resources	medicines, AVICENA tools,
Places	Geometric position of all objects
Actors	Jordi, Barbara
Social Context	
Social interpretation	Patient in bad health, care giver trusted, family loved
Roles	Patient, father, care giver, daughter
Norms	Patient should comply to treatment Care giver must support patient and respect autonomy of patient Family should support patient Doctor is obliged to try to keep patient alive
Activities	Take medicine, give advice, comfort patient,...
Plan patterns	Comfort patient **before** give medicine Give advice **before** leaving
Meaning	Preserve or regain health
Competences	• Domain knowledge and skills: know medicines • Coordination skills : know when to consult **Choice/deliberation skills:** • When health bad consult doctor • When patient refuses medicine start enquiring why • When doctor advices care giver needs to be able to explain advice • ...

Fig. 3. social practices

– *Physical Context* describes elements from the physical environment that can be sensed:
  - *Resources* are objects that play a role in the practice such as medicines, wheel chair, water, table and bed in the scenario.
  - *Places* indicates where all objects and actors are located relatively to each other, in space or time.
  - *Actors* are all people and autonomous systems involved, that have capability to reason and (inter)act.

- *Social Context* contains:
  - *Social Interpretation* determines the social context in which the practice is used.
  - *Roles* describe the competencies and expectations about a certain type of actors.
  - *Norms* describe the rules of (expected) behaviour within the practice.
- *Activities* indicate the normal activities that are expected within the practice. Not all activities need to be performed! They are meant as potential courses of action.
- *Plan Patterns* describe usual patterns of actions defined by the landmarks that are expected to occur.
- *Meaning* refers to the social meaning of the activities that are (or can be) performed in the practice. Thus they indicate social effects of actions
- *Competences* indicate the type of capabilities the agent should have to perform the activities within this practice.

Looking at the characteristics of social practices as given in Figure 3 one can notice some resemblance to the aspects that also play a role in agent organization models (see e.g. [10]). This list can be seen as an analogue of the connection between imposed and emerging norms. Both organizations and social practices give a kind of structure to the interactions between agents. However, organizations provide an imposed (top-down) structure, while the social practices form a structure that arises from the bottom up. Thus where organizational interaction patterns indicate minimal patterns that agents should comply with, the patterns in a social practice indicate minimal patterns that can and are usually used by the agents.

3.4 social intelligent systems

As we argued above socially intelligent agents should use motives, social identity and social practices. Although we will not develop a complete agent architecture for socially intelligent agents we sketched some preliminary ideas in [11] where we combine the different aspects. What is important to mention here is that social practices provide a number of triggers that can be checked in the environment such as the time of day, the location, people and available objects. Those physical elements determine whether a social practice is relevant. If so, it can be started and used as a template context in which the agent finds the possible actions, roles, norms and expectations to follow. If any of the parts is not filled in or gives rise to choices the agent will get into its deliberation cycle in order to fill in the choices.

The social identity of an agent plays a major role in two ways. The different parts of the social identity of an agent all correspond to a set of social practices that are normally shared within a group or are seen as ideal behavior according to a stereotype identity. Thus when a person is in a context where a social identity part is prominent (e.g. family membership when being at home with all family) he will check the social practices pertaining to this social identity.

The second way the social identity plays a role is that when a person identifies a certain social practice to be relevant he will choose his own role in that practice depending on what he expects his social identity will dictate. Thus a family member of the patient with no meical expertise might prefer to play the family role in the practice rather than the care giver role, because he is not sure whether he will have all competences that would be needed for that role.

Where social practices tie into the reactive side of the agent, being triggered by some elements of the environment, the motives can drive the agent to seek out particular situations that would possibly fulfill that motive. Thus if the need of affiliation is high the agent can try to connect to his friends or family and this move might then lead him to a situation in which he can apply a social practice. In our scenario this can be seen when a family member goes visit a patient and when arriving at the patient noticing that he needs to take his medicine. Whether the family member then takes up the role of care giver or as family member depends on the experiences in this situation. If the patient gets very irritated and does not take the medicine when adviced, the family member might try more subtle ways to attract the attention of the patient to the medicine and act more as family than care giver.

4 SAwICENA

To motivate how concepts of socially intelligent systems can be applied to *AVICENA*we introduce a representative scenario. Jordi is a 75 year old widower from Barcelona who has three children. The younger one (Barbara) lives in Barcelona, the middle one (Ana) in Amsterdam and the older one (Patricia) in Paris. Jordi is enrolled in the $AVICENA$ platform, so he has an electronic pill dispenser for supporting his treatment adherence. Jordi's daughters are responsible for re-filling the pill dispenser when new medication doses are required and taking the patient to the doctor for regular health checks and treatment updates. Jordi spends time with his three daughters visiting them for fourth months each in their respective cities BCN, AMS and PAR where he has a doctor assigned. The patient travels with an electronic health record so the different doctors can update it, keeping track of his state. E-prescription systems are available in BCN and AMS but not in PAR. Therefore legal situations must be considered to allow a smooth transition between the health-care system of the different cities, accounting both legal and technological issues.

The above scenario requires a complex institutional or organizational implementation. This can be modelled in *AVICENA*, but we only refer to this in as far as it pertains to the social aspects of the scenario. First of all, it is clear that Jordi wants to be with all his daughters regularly. Thus his affiliation motive seems to be an important driver for his behaviour.
The daughters have two social identities (related to the scenario), they are both daughters and care givers. With respect to the first identity there is a strong norm that one has to respect and obey one's parents. As a parent, Jordi does not want to be dependent on his children, because as a parent one has to pro-

vide for one's children, take care of them, etc. However, the social identity of the daughters as being a care giver does give them the responsibility to take care of their father's health. This might lead to a situation where they have to give him orders with respect to taking his medication. Thus we see a tension between the two identities.
The tension can be resolved in an organisational way by appointing professional care givers only for the care giver role. However, this is not very cost efficient and even sometimes impossible due to the fact that Jordi moves around every four months.

We use the social practices to analyse the whole scenario. The routine Jordi has to visit each of his daughters in turn every four months can be seen as a social practice. This social practice stretches over the different locations in Barcelona, Paris and Amsterdam. and the actors involved are Jordi and his daughters. The social interpretation of the social practice is that the father loves his daughters and shows his devotion by visiting them in turn for equal length. The daughters show their love for their father by hosting him for those four months. Thus the social meaning of the practice is to express the status of each in the family that is spread out over Europe. The roles are the father and the daughter role. The norms are that the father will provide for himself as much as possible, that the daughters involve their father in their family life, that the father commits to follow the round robin visits. The activities can be given as very general visiting and interacting of Jordi with his daughters. The plan pattern is just the round robin nature of the visits. The meaning of the whole social practice is to show the family ties and strengthen them. The competences expected are minimal. Jordi should have some financial means to travel and maybe contribute to the staying costs. The daughters should have the competence to cope with their father.

The next step is to tie all these elements into the scenario where the daughters are somehow co-responsible for the treatment of their father and check whether he takes his medicines.
We have established that the father has an intrinsic motive to visit his daughters. The social practice establishes a practical way of realizing this. If we want the *AVICENA* system to support the family such that Jordi will take his medicines at the right time it should connect with this social practice. A simple way to force this is to connect the medicine dispenser to the electronic patient file. While the medicines are dispensed in the correct dose on the right days and times nothing is reported in the electronic patient file. However, whenever there is a deviation this can be marked in the file. If the electronic patient file has several of these marks it might signal this fact and forbid the patient to travel due to health risks. Thus this event will disrupt the social practice. Indirect followig the treatment correctly now becomes tied to showing his love to his daughters and is motivated by his affiliation motive. Thus Jordi gets an internal motivation that is in line with his behaviour and makes him aware of the medicines not only from a health perspective, but also from a family perspective.

The above shows already the use of the social aspects in designing the support system. We could also go one step further and include the social aspects in the

agents that are part of the *AVICENA*platform. Given that these agents would have an understanding of their role and the role of all the humans in this scenario they can support the patient by aligning their actions with the social practices of the patient. In the above we used the very large social practice of visiting the daughters for a few months. However, their are also daily practices that can be used to combine with dispensing medicines. E.g. with dinner or when the daughter checks in with her father. In that way the visit of the daughter every day becomes combined with taking medicines. This in itself will make it easier for the daughter to remind her father to take the medicines, because it has become part of the visit to take the medicines.

We have given some very preliminary sketches to show the added value of incorporating social aspects in these complex socio-technical systems, but it already indicates its potential at different levels.

5 Related Work

Assistive Technologies (AT) can be effectively used for guiding elderly with their prescribed treatments, avoiding major problems such as non-compliance with the treatment and adverse drug reaction. There exists a range of different technological approaches, from the use of smart devices by patients (such as smart pill dispensers [12]) to Ambient Intelligence [1] [26] (AmI) environments supporting independent living. The specific area of health monitoring devices is currently characterised by application-specific and hardware-specific solutions that are mutually non-interoperable and are made up of diverse architectures [30]. Furthermore, systems mainly focused on activity monitoring and reminders tend to be rejected by end users, who may end up feeling that the system becomes too intrusive on their privacy [23]. Research on smart home environments and Ambient Assisted Living is moving towards a more holistic view, trying to create not only patient-centric AmI solutions, but also connecting the patient with other relevant actors in their medical treatments or event connecting patients to avoid isolation and depressive attitudes. In the rest of the section we will focus on some agent-oriented AmI solutions that are close to the work presented in the paper.

The GerAmi project [8] creates a networked AmI solution where agents are used to enhance communication and work scheduling, effectively making profesional caregivers' working hours more productive. Based in the THOMAS organizational architecture [4], roles, organizational units and norms have been modelled. However, none of the articles explaining the THOMAS architecture analysed so far includes a clear example of such organizational definition, or how norms are operationalised. Furthermore, social concepts such as social identity, social realtions, values or social practices are not present in the framework.

$COMMODITY_{12}$ [18] focuses on providing advice, recommendations and alerts to diabetic patients based on their data, and at the same time assist medical personnel, who is in charge of these patients, facilitating informed and timely decisions. The system consists in two main components: first, a set of

devices that collect health-related data (e. g., activity and body signals). Second, a set of personal agents with expert biomedical knowledge that interpret the data via a reasoning process to generate a high level representation of patient's health status. These interpreations are then provided to relevant actors in the scenario (e. g., patients and health care professionals) in the form of feedback reports. The main idea is integrating sensors, intelligent agents, knowledge bases and users within a single system. The work introduces the $\mathcal{LAMA}$ architecture for developing software agents that can reason about a medical domain. Agents are deployed using the GOLEM agent platform [5]. Unlike other approaches analysed (e. g., GerAmi and *AVICENA*) $COMMODITY_{12}$ does not explicitly define the social structure where agents and devices operate. In $COMMODITY_{12}$ norms are reflected implicitly in the behaviours of the agents. Furthermore, the representation of the social context in $COMMODITY_{12}$ is not explicit but recent research[19,20] demonstrates it can be acquired through lifestyle activity recognition of patient's interaction with the system.

In [2] a system for automated real-time monitoring of medical protocols is proposed. The system consists on two main components. First, a domain-independent language for protocol specification, accompanied by a user-friendly specification tool that that allows health care experts to model a medical protocol and translate into the systems protocol specification language. Second, a semi-autonomous system that understands the protocols and supervises their application. Medical services are modelled as agents, and a medical protocol is interpreted as a negotiation process between agents. The system is able to observe the negotiation, effectively warning about forbidden actions and decisions. The system is applied to health care environments where every staff person plays one or more roles. A role specifies a particular service (e. g., infirmary, surgery, *etc.*) and a medical protocol specifies possible interactions between the different services in front of a particular pathology. The protocol can suggest or forbid medical decisions depending on the medical history and evolution of the patient. Agent interactions are performed as message exchanges through a communication layer. *Supervisor agents* track such interactions and validate them. Suggested actions correspond to medical guidelines and forbidden actions to medical protocols. However, the social model is too protocol-driven, and there are no way to model important issues such as, e.g., the patients' motives.

Robot ecologies [27] are a growing paradigm in agent-based AmI in which several robotic systems are integrated into a smart environment. Such systems hold great promises for elderly assistance. Robocare [6] is a project deployed on a domestic test-bed environment that combines a tracking component for people and robots and a task execution-supervision-monitoring component. The system is composed of several software and hardware agents, each providing a set of services, and an event manager that processes requests to the different services and directs them to the appropriate agents. The system also includes a monitoring agent, with knowledge of the assisted person's usual schedule. However, agent coordination and monitoring are heavy computational processes, limiting the tested scenarios to only 2-3 persons and only a small portion of the do-

mestic environment. the ILSA (Independent LifeStyle Assistant) project [15], that passively monitors the behaviours of the inhabitants of the residential laboratory, alerting relatives in case of potentially dangerous situations (*e.g.*, the user falls). ILSA presents two main innovations with regards to the Robocare project: 1) Agents autonomously interact within them in order to achieve their goals, without the need of an event manager agent that coordinates them (but a centgralized coordination agent is used to transform context-free perceptions provided by the agents into context-aware perceptions); and 2) Agents are able to learn schedules based on the daily tasks performed by the inhabitants. However, once a schedule has been learned, the user is not able to deviate from it without raising an alarm. Focus in both systems is on activity monitoring and the coordination between the human and the artificial devices, and thus other social aspects such as the patients' relationship with caregivers are not part of the model.

An interestingly rich model is the the AOE^2 framework presented in [7]. AOE^2 integrates (in a model that is both general and coherent) the main concepts to be considered in order to build an agent-based simulator for the particular domain of health care. It is able to reproduce the behaviour of the social system by presenting the decision making entities of the studied system as agents. The main idea behind the AOE^2 framework is focusing in high level conceptual issues regarding health care model development process, while offering a guideline for carrying out this process independently of technical choices. The idea of applying a framework to agent-based simulations in the healthcare domain is appealing. The complexity and dynamics of the domain (e. g., the high degree of uncertainty inherent to clinical processes, the involvement of multiple distributed service providers and decision makers, *etc.*) make it useful for applying agent-based simulations. Furthermore, the approach is also valid for providing a tool able to asses the possible outcomes of the different actions that can be taken in order to improve the system, making it more efficient or sustainable from an economic point of view. However the model does not include mental models of the individuals' motives, values and social identities, thus being unable to tackle the informal relations that we are trying to model in our work.

6 Conclusion and Future Work

In this paper we have shown the potential of extending the *AVICENA*system with social intelligence. We have outlined with social aspects seem of particular importance. I.e. social motives, social identity and social practice. We have sketched their role in the agent deliberation and have shown their use both in the design of a socially intelligent system as well as how individual agents could profit from these social enhancements.

Of course, this paper only gives some preliminary steps and one of the first steps to take is to give a more formal representation of the social aspects such that we can give a more precise and formal account of their influence on the

agent deliberation. We hope to do some of this work while actually starting on an implementation of the scenario in *AVICENA*.

A second important step is to describe the relations between all these different aspects in an agent deliberation not just for particular scenarios but also in a more generic way. I.e. do agents always start with social practices and then decide on actions based on their motives or decide upon their roles in the social practice based on their identity? Or do they start with their identity and find social practices fitting with that identity? Or better still is their no fixed order but is that determined by the situation?

As can be seen there are many interesting issues that should be looked into, but this paper shows at least that they are issues worth investigating.

References

1. Acampora, G., Cook, D.J., Rashidi, P., Vasilakos, A.V.: A survey on ambient intelligence in healthcare. Proceedings of the IEEE 101(12), 2470–2494 (2013)
2. Alsinet, T., Ansótegui, C., Béjar, R., Fernández, C., Manyà, F.: Automated monitoring of medical protocols: a secure and distributed architecture. Artificial Intelligence in Medicine 27(3), 367–392 (2003)
3. Álvarez-Napago, S., Cliffe, O., Padget, J.A., Vázquez-Salceda, J.: Norms, Organisations and Semantic Web Services: The ALIVE approach. Workshop on Coordination, Organization, Institutions and Norms at MALLOW'09 (2009)
4. Bajo, J., Fraile, J.A., Pérez-Lancho, B., Corchado, J.M.: The THOMAS architecture in home care scenarios: A case study. Expert Systems with Applications 37(5), 3986–3999 (2010)
5. Bromuri, S., Stathis, K.: Situating cognitive agents in golem. In: Engineering environment-mediated multi-agent systems, pp. 115–134. Springer (2008)
6. Cesta, A., Oddi, A., Smith, S.F.: A Constraint-Based Method for Project Scheduling with Time Windows. Journal of Heuristics 8, 109–136 (2002), http://dx.doi.org/10.1023/A:1013617802515
7. Charfeddine, M., Montreuil, B.: Toward a conceptual agent-based framework for modelling and simulation of distributed healthcare delivery systems. CIRRELT (2008)
8. Corchado, J.M., Bajo, J., Abraham, A.: GerAmi: Improving healthcare delivery in geriatric residences. Intelligent Systems, IEEE 23(2), 19–25 (2008)
9. Dignum, F., Prada, R., Hofstede, G.: From autistic to social agents. In: AAMAS 2014. pp. 1161–1164 (May 2014)
10. Dignum, V.: A Model for Organizational Interaction: based on Agents, founded in Logic. SIKS Dissertation Series 2004-1, Utrecht University (2004), phD Thesis
11. Dignum, V., Dignum, F.: Contextualized planning using social practices. In: Coordination, Organisations, Institutions and Norms in Agent Systems X. LNAI, vol. 9372. Springer (2015)
12. Georgia Institute of Technology: Aware Home Research initiative. Tech. rep., Georgia Institute of Technology (2012), http://www.cc.gatech.edu/fce/ahri/projects/index.html.
13. Gómez-Sebastià, I.: NoMoDei: A framework for Norm Monitoring on Dynamic electronic institutions. Universitat Politecnica de Catalunya (2016), phD Thesis

14. Gómez-Sebastià, I., Garcia-Gasulla, D., Álvarez-Napagao, S., Vázquez-Salceda, J., Cortés, U.: Towards an implementation of a social electronic reminder for pills. VII Workshop on Agents Applied in Health Care (2012)
15. Haigh, K.Z., Kiff, L.M., Myers, J., Guralnik, V., Geib, C.W., Phelps, J., Wagner, T.: The Independent LifeStyle Assistant (I.L.S.A.): AI Lessons Learned. In: In The Sixteenth Innovative Applications of Artificial Intelligence Conference (IAAI-04. pp. 25–29 (2004)
16. Holtz, G.: Generating social practices. JASSS 17(1), 17 (2014), http://jasss.soc.surrey.ac.uk/17/1/17.html
17. Istepanian, R., Laxminarayan, S., Pattichis, C.S.: M-health. Springer (2006)
18. Kafalı, Ö., Bromuri, S., Sindlar, M., van der Weide, T., Aguilar Pelaez, E., Schaechtle, U., Alves, B., Zufferey, D., Rodriguez-Villegas, E., Schumacher, M.I., et al.: Commodity 12: A smart e-health environment for diabetes management. Journal of Ambient Intelligence and Smart Environments 5(5), 479–502 (2013)
19. Kafalı, Ö., Romero, A.E., Stathis, K.: Activity recognition for an agent-oriented personal health system. In: PRIMA 2014: Principles and Practice of Multi-Agent Systems, pp. 254–269. Springer (2014)
20. Luštrek, M., Cvetkovic, B., Mirchevska, V., Kafalı, Ö., Romero, A.E., Stathis, K.: Recognising lifestyle activities of diabetic patients with a smartphone. In: Proceedings of Pervasive Health 2015 : Workshop on Personal Health Systems for Chronic Diseases (to be puslished)
21. McClelland, D.: Human Motivation. Cambridge Univ. Press (1987)
22. National Council on Patient Information and Education.: Enhancing Prescription Medicine Adherence: A National Action Plan. Tech. rep., National Council on Patient Information and Education. (2007)
23. Niemelä, M., Fuentetaja, R.G., Kaasinen, E., Gallardo, J.L.: Supporting Independent Living of the Elderly with Mobile-Centric Ambient Intelligence: User Evaluation of Three Scenarios. In: AmI. pp. 91–107 (2007)
24. Population Division UN Department of Economic Social Affairs: Population ageing and development: Ten years after Madrid. Tech. Rep. 2012/4, Population Division UN Department of Economic Social Affairs (Dec 2012)
25. Reckwitz, A.: Toward a theory of social practices. European Journal of Social Theory 5(2), 243–263 (2002)
26. Sadri, F.: Ambient intelligence: A survey. ACM Computing Surveys (CSUR) 43(4), 36 (2011)
27. Saffiotti, A., Broxvall, M., Gritti, M., LeBlanc, K., Lundh, R., Rashid, J., Seo, B., Cho, Y.J.: The PEIS-ecology project: vision and results. In: Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on. pp. 2329–2335. IEEE (2008)
28. Schäfer, G.: Europe in figures. Eurostat statistical yearbook (2008)
29. Shove, E., Pantzar, M., Watson, M.: The Dynamics of Social Practice. Sage (2012)
30. Vermesan, O., Friess, P.: Internet of Things: converging technologies for smart environments and integrated ecosystems. River Publishers (2013)
31. World Health Organization: Adherence to long-term therapies. Evidence for action. Tech. rep., World Health Organization (2003)

Using Petri Net Plans for Modeling UAV-UGV Cooperative Landing

Andrea Bertolaso and **Masoume M. Raeissi** and **Alessandro Farinelli** and **Riccardo Muradore** [1]

Abstract. Aerial and ground vehicles working in corporation are crucial assets for several real world applications, ranging from search and rescue to logistics. In this paper, we consider a cooperative landing task problem, where an unmanned aerial vehicle (UAV) must land on an unmanned ground vehicle (UGV) while such ground vehicle is moving in the environment to execute its own mission. To solve this challenging problem we consider the Petri Net Plans (PNPs) framework, an advanced planning specification framework that allows to design and monitor multi robot plans. Specifically, we use the PNP framework to effectively use different controllers in different conditions and to monitor the evolution of the system during mission execution so that the best controller is always used even in face of unexpected situations. Empirical simulation results show that our system can properly monitor the joint mission carried out by the UAV/UGV team, hence confirming that the use of a formal planning language significantly helps in the design of such complex scenarios.

1 Introduction

Use of cooperative unmanned air and ground vehicles has been growing rapidly over the last years, search and rescue[8], target detection and tracking[21] and mines detection and disposal [22, 3] are a few examples of such applications that benefit from collective behavior of different types of unmanned robots. As in any multi robot system, a variety of cooperative scenarios can be imagined in different applications: aerial robots assist ground robots (Aerial robots can provide the ground robots with information related to the environment, ex. landmark maps), ground robots assist aerial robots or ground and aerial robots cooperate to achieve a task (for example exploration and surveillance or target detection and tracking tasks)[15].

Surveying the relevant literature, UAV/UGV corporation has been addressed from different perspectives. One research direction in this area is the development of controlling schemes that provide control laws for the different vehicles, while considering that their motion must be coordinated. For example, Brandao and colleagues, in [2], provide a decentralized control structure that involves an helicopter and a team of UGVs to accomplish a 3D-trajectory tracking mission. Similarly, the approach proposed by Owen and colleagues [17] aim at developing a coordinated system where UAVs and UGVs must track a dynamic target.

Another strand of research focuses on cooperative path planning and task assignment methods for systems composed of UAVs and UGVS. For example, Yu and colleagues [21] focus on path planning for cooperative target tracking, while Dewan and colleagues [6] consider coordinated exploration for a UAVs/UGVs team by using a task assignment solution approach based on integer programming.

The main focus of such previous work is on acquiring and integrating data gathered by each vehicles to perform tasks such as exploration, surveillance or target tracking. Here we turn our attention to a cooperative control scenario, where the UAV/UGV team should operate in tight cooperation to perform a joint task. In particular, here we focus on a cooperative landing scenario, where the UAV must land on the UGV while the UGV is moving in the environment to execute its own mission. Our goal is for the UAV to perform a fast and safe landing maneuver, hence we propose a strategy where the UAV quickly approaches the UGV and then carefully plans a safe landing trajectory. In this context, by tight coordination we mean that robots must continuously synchronize their individual actions to successfully perform the joint task. This is because the joint task imposes execution constraints to a vehicles that might depend on the state of the other vehicle. For example, in our landing scenario, the UAV must know the intended future locations of the UGV to properly plan a trajectory so to smoothly land on the UGV. This tight cooperation is in contrast with loose cooperation, where robots can execute their individual actions in isolation but should coordinate (and communicate) only at key points. For example, when exploring a region robots should avoid overlapping too much but once they decided their area of competence they do not need a continuous communication with the other platforms.

Now, a crucial open issue for multi robot system that must perform tight cooperation is to recover from possible failures due to unexpected events. For example, consider a situation where the UAV is initiating the landing maneuver based on the future positions communicated by the UGV. If the UGV must suddenly change its current trajectory (i.e., due to a moving obstacle) the UAV should smoothly adapt its plan to recover from a possible failure.

In this paper we investigate the use of high level language or team plans [20, 11, 23, 7] to describe member's actions and to monitor the activities of vehicles during mission execution so to achieve the collective behaviors and goals even in face of such unexpected events. Specifically, we focus on Petri Nets and related approaches which build on PNs structure (e.g., Petri Net Plans [23] and Colored Petri Nets [9]), which have proved to be excellent tools for modeling multi robot systems.

In more detail, the main contribution of this paper is to investigate the use of the Petri Net Plan (PNP) specification framework to specify the collaborative landing task. There are several benefits related to the use of the PNP framework: first it provides a rich graphical representation that helps the designers to create plans with minimal effort, second the generated plans can be monitored during the execution, third PNPs support well-defined structures for handling tight

[1] Computer Science Department, University of Verona, Italy, email: andrea.bertolaso@studenti.univr.it, masoume.raeissi@univr.it, alessandro.farinelli@univr.it, riccardo.muradore@univr.it

coordination and on-line synchronization in multi robot systems.

In summary, this paper makes the following contributions to the state of the art:

- we use an advanced framework for multi agent plan specification to design a complex cooperative behavior in multi robot systems. Specifically, we design an effective strategy for cooperative landing for our UAV/UGV system that is able to recover from unexpected situations (i.e., sudden deviation of the UGV from the planned trajectory). To the best of our knowledge this is the first application of a team-oriented plan specification framework to a complex cooperative control scenario such as cooperative landing.
- We evaluate our approach in a realistic simulation environment using state of the art tools for robot control and simulation. Specifically, we use ROS to connect and control the simulated platforms, and V-REP to simulate the two platforms and the environment. Our experiments show that the proposed approach can effectively monitor the cooperative behavior of the two vehicles recovering from possible failures. Specifically a video (reference) shows an exemplar execution of our framework describing the different operations carried out by the vehicles as well as the different states of the monitoring framework.

The remainder of the paper is organized as follows: Section 2 describes the related work on cooperative UAV/UGV applications and team oriented plans. Section 3 provides necessary background on the PNP framework while Section 4 detail our cooperative control strategy and the plan we designed to monitor the mission. The evaluation and simulation setups are explained in section 5. Section 6 concludes the paper and outline possible future research directions.

2 Related Works

In this section we will first discuss previous works on UAV/UGV cooperation applications and then Petri Net Plans framework will be described in more details.

2.1 UAV/UGV Cooperation

There are a wide variety of applications that take advantage of cooperative multi vehicle team including aerial and ground vehicles. To model and solve the cooperation tasks, several multi vehicle platforms have been proposed and investigated for different applications. Each platform is characterized by the path planning algorithm and the task assignments method. Yu *et al.* model a tracking problem using UAVs/UGVs cooperation based on the probability of the target's current and predicted locations [21]. The path planning algorithm is designed to generate paths for a single UAV or UGV maximizing the sum of probability of detection over a finite look-ahead. Dewan *et al.* propose an exploration strategy for coordinated unmanned ground vehicles (UGV) and micro-air vehicles (MAV) [6]. The exploration is modeled as an Integer Programming optimization problem.

UAV/UGV collaboration can be also exploited for mine detection [22]. The UGV navigates to the Unexploded Ordnance (UXO) positions based on the data sent by UAV. Cantelli *et al.* propose an architecture to allow cooperation between a ground robot and a quadrotor UAV [3]. The UAV can autonomously follow the ground robot, by using an image processing algorithm: aerial images are used to plan trajectories via a developed webGIS platform.

Phan and Liu propose a hierarchical 3-layered UAV/UGV cooperative control framework for a wild fire detection scenario [19]. The model consists of a mobile mission controller, which is the generic mission planner (based on the defined autonomy), and two particular vehicle platforms which can optimally run the designed plans. Compared to our work, their framework does not monitor the execution of the plan and there is no plan validation tool: both these features are provided by the PNPs framework.

The most similar research to our work from the problem definition perspective can be found in [5] where an autonomously coordination for the landing between a quadrotor UAV and skid-steered UGV is proposed. A joint decentralized controller is designed on top of local nonlinear controllers that linearize the mathematical model of each vehicle via feedback. Once the vehicles are spatially close enough to each other, an automated landing procedure for the quadrotor is activated. This procedure is based on a tracking controller for the quadrotor altitude state.

In this paper we focus on a high level planning language for modeling the cooperative behavior instead of implementing cooperative perception techniques. We use PNPs that allow to handle external events and interruptions of the execution of operations. This property has great influence on keeping the team's goal integrated in tightly coordinated tasks.

2.2 Team plans

The problem of monitoring plan execution in multi robot systems is a key issue when such systems must be deployed for real- world applications, where the environment is typically dynamic and action execution is non -deterministic. Two successful BDI-based frameworks for plan specification are STEAM and BITE, which enable a coherent teamwork structure for multiple agents. The key aspect of STEAM [20] is team operators, which are based on the Joint Intentions Theory introduced by [4]. In STEAM, agents can monitor the team's performance and reorganize the team based on the current situation. BITE, which was introduced by Kaminka & Frenkel [12], specifies a library of social behaviors and offers different synchronization protocols that can be used interchangeably and mixed as needed. However, while both these works provide key contributions for building team oriented plans, they do not provide any specific mechanism for interrupting the execution of such plans. There is substantial literature on the topic of using Petri Nets [18] and variants such as Colored Petri Nets [10] as the basis for representing team plans. Similar to state machines and other directed graphs, Petri Nets give an intuitive view of the plan, but provide additional power useful for multi robot teams, such as synchronization and concurrency. Significant work has produced Petri Net analysis tools [16] [1] which can determine many of its behavioral properties, such as reachability, boundedness, liveness, reversibility, coverability, and persistence. For complex team plans, these automated methods for finding errors before testing them on simulated or physical platforms is an important strength. [23] proposed an approach for plan monitoring called Petri Net Plans (PNPs). PNPs takes inspiration from action languages and offers a rich collection of mechanisms for dealing with action failures, concurrent actions and cooperation in a multi robot context. One important functionality offered by the formalism of PNP is the possibility to modify the execution of a plan at run-time using interrupts.

3 Background: Petri Net Plans (PNPs)

Petri Net Plans (PNPs) is a framework for designing, representing and executing complex multi robot behaviors. The syntax and the semantics of PNPs is based on Petri Nets (PNs) [16]. In addition

to supporting PNs properties, it is equipped with several features. For example, in order to provide cooperative behaviors in robotic application, different kind of operators are defined by the framework, such as the *coordination operator* and the *interrupt operator*. In the following we discuss the structure of PNP language in more details clarifying the use of the different operators and strictures in our plan.

In general a PNP is a PN$\langle P, T, F, M_0 \rangle$ with a domain specific interpretation and an extended semantics.

Specifically, a Petri Net is represented by a directed bipartite graph, in which nodes could be either places or transitions, arcs connect places to transitions and vice versa. Places in a Petri net contain a discrete number of marks called *tokens*. A particular allocation of tokens to places is called a *marking* and it defines a specific state of the system that the Petri Net represents. In more detail, the PN tuple is formed by a finite set of places $P = p_1, p_2, ..., p_m$ and a finite set of transitions $T = t_1, t_2, ..., t_n$, where $P \cup T \neq O$ and $P \cap T = O$. Places and transitions are connected by a set of edges $F \subseteq (P \times T) \cup (T \times P)$[2]. Finally, an initial marking $M_0 : P \to 0, 1$ specifies the initial distribution of tokens over the PNP. Notice, that while in general Petri Nets the initial distribution of tokens consider a positive, integer number of tokens for each place in PNP we restrict this to zero or one token. This is because, in PNP tokens define execution threads for the robot's actions, hence there should not be two tokens in the same place.

In a PNP, there are four different type of places: $P = P^I \cup P^O \cup P^E \cup P^C$, where:

- P^I is the set of input places, which model initial configurations of the PNP;
- P^O is the set of output places, which model final configurations of the PNP;
- P^E is the set of execution places, which model the execution state of actions in the PNP;
- P^C is the set of connector places, which are used to connect different PNPs.

Also transitions are partitioned in three subsets $T = T^S \cup T^T \cup T^C$, where:

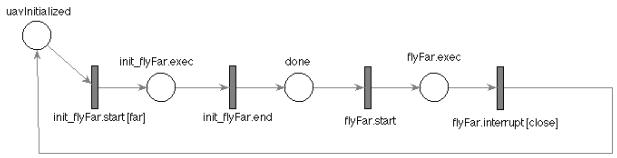
- T^S is the set of start transitions, which model the beginning of an action/behavior;
- T^T is the set of termination transitions, which model the termination of an action/behavior;
- T^C is the set of control transitions, which are part of the definition of an operator.

Two types of actions are considered in PNPs framework: ordinary and sensing actions. Ordinary actions are deterministic non-instantaneous actions. For example in figure 1(a) which is part of our petri net model, *init_flyFar* is an ordinary action. Since it consist of a sequence of start event *init_flyFar.start [far]*, execution state *init_flyFar.exec*, and termination event *init_flyFar.end*.

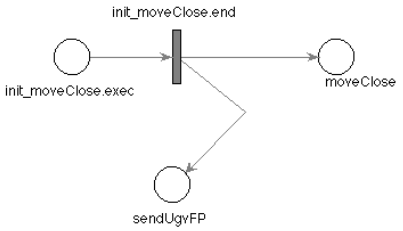
In contrast to the ordinary actions, sensing actions are non-deterministic which means that the outcome of the action may be specified at execution time.

We can build more complex operators by combining different PNPs structure. The most important operators to build complex PNPs are:

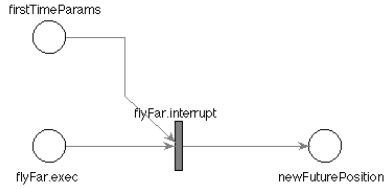
[2] Notice that, standard Petri Nets include also a function, that associate a weight to each edge specifying the number of tokens that are required by the transition to fire (when the edge goes from a place to a transition) or the number of tokens that are inserted in the place (when the edge goes from a place to a transition). In PNP the labels are all 1 hence we do not include the weight function here.



(a) Interrupt operation



(b) Fork operation



(c) Join operation

Figure 1. PNPs different operators has been applied in cooperative UAV/UGV Petri Net model

Interrupt Operator Interrupt operator, is a very powerful tool for handling action failures. In fact, it can interrupt actions upon failure events and activate recovery procedures. The plan shown in figure 1(a) shows an interrupt operator where transition *flyFar.iterrupt [close]* will interrupt the execution of *flyFar* action when the *close* condition happens in the system.

Fork operator Figure 1(b) shows an example of fork operator which indicates that after firing the *init_moveClose.end* transition, the token inside place *init_moveClose.exec* will go to both places *moveClose* and *sendUgvFP*. Actually the fork operator in PNP framework generates multiple threads from one thread.

Join Operator Figure 1(c) illustrate a join structure in the created plan. This operator provides the simultaneous execution of multiple

threads or tokens.

The PNP framework has been successfully applied on several robotic platforms and in different domains and is available at `https://sites.google.com/a/dis.uniroma1.it/petri-net-plans/`

4 Problem Description: UAV/UGV Cooperative Landing Scenario

The problem addressed in this paper is a particular kind of collaboration between heterogeneous autonomous vehicles: the landing of an UAV on an UGV. We model the execution of this task by exploiting the power of the PNP framework discussed in Section 3. The collaboration task is composed of three phases:

1. both the UGV and UAV are moving according to their specific and non-cooperative tasks;
2. the UAV approaches the UGV (*flyFar* action using the PNP terminology);
3. the UAV lands on the UGV (*flyClose* action using the PNP terminology).

In Phase 2 the UAV is using its sensing system (e.g. camera) to locate the UGV and plans the faster trajectory to approach the UGV. In this phase the UGV in not aware of the intention of the UAV and so it is continuing its task as in Phase 1.

In Phase 3, the UAV is close to the UGV and information are exchanged between them: the UGV is getting aware of the intention of the UAV and so it decreases its velocity and sends to the UAV its planned trajectory to easier the landing. This means that the UGV is still pursuing its objective (e.g. patrolling an area) but in a slower way.

The key element in Phases 2 and 3 is the efficient generation of the trajectories for the UAV. To generate the trajectory, we used the Type II Reflexxes Motion Libraries [13, 14] which allows to force trapezoidal velocity profiles only. This means that we can set the maximum speeds (v_x^{max}, v_y^{max} and v_z^{max} along x, y and z, respectively) and the maximum accelerations (a_x^{max}, a_y^{max} and a_z^{max}). In this work, we assume that only the Cartesian positions have to be computed, i.e. $x(\cdot), y(\cdot)$ and $z(\cdot)$, whereas the yaw angle is set in such a way to make the UAV always pointing towards the UGV.

During Phase 2, the UAV knows its current position $\mathbf{p}_{UAV} = \{x(t), y(t), z(t)\}$, velocity $\mathbf{v}_{UAV} = \{v_x(t), v_y(t), v_z(t)\}$, and the actual position of the UGV $\mathbf{p}_{UGV} = \{X(t), Y(t)\}$. For the UGV the coordinate along z is not important. The UAV needs only to know the height $\bar{Z}$ of the area with respect to the ground where it is supposed to land, i.e. $Z(t) = \bar{Z}, \forall t$.

The following excerpt of the code explains how we used the Type II Reflexxes Motion Libraries

```
ReflexxesAPI *RML = new ReflexxesAPI(NDoF,Ts);
for (int i = 0; i < NDoF; i++)
{
input_params->CurrentPositionVector->VecData[i] =
                                uavMotion.position[i];
input_params->CurrentVelocityVector->VecData[i] =
                        uavMotion.currentVelocity[i];
input_params->MaxVelocityVector->VecData[i] =
                             uavMotion.maxVelocity[i];
input_params->MaxAccelerationVector->VecData[i] =
                        uavMotion.maxAcceleration[i];
input_params->TargetPositionVector->VecData[i] =
                                    ugvCurrentPos[i];
input_params->TargetVelocityVector->VecData[i] = 0;
input_params->SelectionVector->VecData[i] = true;
}
error_value = RML->RMLPosition(*input_params,
                        output_params,Flags);
```

The `input_params` structure contains the information about the current and target positions and the kinematic constraints (max speed and acceleration), whereas `output_params` gives the planned trajectory for each degree of freedom (`NDoF`). In the present scenario `NDoF` is equal to three. We selected the *Phase-synchronization* planning mode: this means that the trajectories for x, y end in the target positions $X(t), Y(t)$ at the same instant. During Phase 2, z is kept constant.

In Phase 3, the UAV uses the T-seconds ahead information about the position received by the UGV as target point, instead of the UGV current position of the UGV as in Phase 2.

The previous code changes only in the following line

```
input_params->TargetPositionVector->VecData[i] =
                                        ugvFuturePos[i];
```

where `ugvFuturePos` is provided by the UGV, whereas the time interval T is constant, known to the UAV and sets as a trade-off between promptness and smoothness.

Unlike in Phase 2, the constraint on T implies that we have to tune the maximal velocity (along x, y, z) in a way that the UAV will be in $X(t+T), Y(t+T), \bar{Z}$ at $t+T$, i.e. not later but also not before.

To guarantee such behavior, we have to solve the following problem any time the UAV receives an update from the UGV:

$$\bar{\mathbf{v}}^{\star}_{UAV} = \arg \quad \max_{\bar{\mathbf{v}}^{max}_{UAV}} \quad \|\mathbf{p}_{UAV}(t+T) - \mathbf{p}_{UGV}(t+T)\| \quad \text{subject to} \quad \bar{\mathbf{v}}_{UAV} \leq \mathbf{v}^{max}_{UAV}$$

where $\mathbf{v}^{max}_{UAV} := \sqrt{(v_x^{max})^2 + (v_y^{max})^2 + (v_z^{max})^2}$. It is possible that $\mathbf{p}_{UGV}(t+T)$ cannot be reached by the UAV in T seconds also moving at the maximum speed, i.e. $\mathbf{p}_{UAV}(t+T) \neq \mathbf{p}_{UGV}(t+T)$. In this case the UAV can only move at maximum speed in the right direction. Since in Phase 3 the maximum velocity of the UAV is larger than the maximum velocity of the UGV, there would be a moment where the minimization problem will find out a feasible value for the speed to plan the trajectory that satisfies exactly the final condition. The UGV will then safely land on the UGV in T seconds accomplishing the cooperative task

As it is better explained in the next Section, it is possible that, due to obstacles (e.g. trees, buildings) or other application-dependent reasons, the UAV cannot land as expected.

5 Simulation and Evaluation

Figure 3 represents the team plan has been created to model the above mentioned cooperative task. We used JARP to create this Petri Net plan, however any of the available graphical tool that supports pnml file format could be used. According to Figure 3, the plan consists of a part for controlling UGV's behavior and another part for UAV's behavior.

Actions name and all external conditions have been defined in the plan. Actions represent robot behaviors, for example in our case the *flyFar* action represents the UAV flying towards the UGV constantly following its position. Conditions are external events and need to be checked at run time. The possibility to define conditions is a powerful feature that allows to enrich the plan behavior at run time.

The plan execution will be started by initializing the UGV and then the UAV. Both robots will execute the *moveFar* and *flyFar* actions, until UAV gets close to the UGV or decides to landing. The close and far distances are application-dependent. When the UAV is close

to the UGV, the *flyFar* action is interrupted and UAV sends the *close* event to the UGV. The UGV *moveFar* action is interrupted as well. The part of the plan which is highlighted in figure 3 controls the communication and synchronization between the two vehicles.

When the vehicles are getting close, UAV's behavior should change based on UGV's future position. UAV must be informed of UGV's future position to coordinate their actions. Thus UAV will receive UGV's future position periodically (T seconds). Every time a new future position is sent to the UAV, if it is not the first, this new position will interrupt the *flyClose* action so the UAV can recompute flight trajectory in order to follow the new location of UGV. Basically the if/else block is used to decide whether we should stop the *flyFar* action (which happens only once, when the first new position is received), or the *flyClose* action, which may happen several time (until the UAV gets over the UGV).

To execute and validate the developed plan, a simulation platform running on a Linux operative system is used. The following software tools are integrated to implement and execute the above mentioned plan within the Robot Operating System (ROS) middleware:

- JARP: it is a graphical interface for creating the Petri net plan,
- PNP: it is a library that processes pnml file and executes the plan,
- PNPros: it is the bridge between the PNP library and ROS that allows the execution of PNP in ROS using the actionlib module. It can be used for implementing different actions and for defining the firing rules for transitions,
- V-REP: it is used for the visualizing and simulation of the environment,
- Reflexxes: it is a library that computes the UAV trajectories.

The following steps are required to set up the system using the PNPs library and ROS. When the plan is designed with JARP, we have to hand coding actions and conditions, and makes them available to the PNPros system which connects the PNP library with ROS. Actions will be implemented using the ROS Action-lib interface.

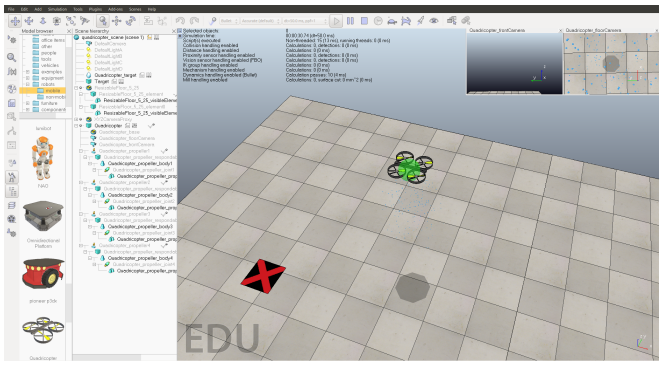
For running the experiments, we create in V-REP a simulation environment containing the UAV and the UGV. The initial position of both vehicles can be chosen arbitrarily in order to obtain different experimental setups. Communication with V-REP is possible through ROS topics. When the simulation environment and the system that handles the plan are launched, the initial position of UAV and UGV is retrieved from V-REP via ROS topics. With these information the plan can start its progress by using PNP library. PNP library communicates with PNPros to

- start new actions (a thread is launched for each action),
- check external conditions based on the environmental knowledge that is available thanks to PNPros, and
- interrupt a running action.

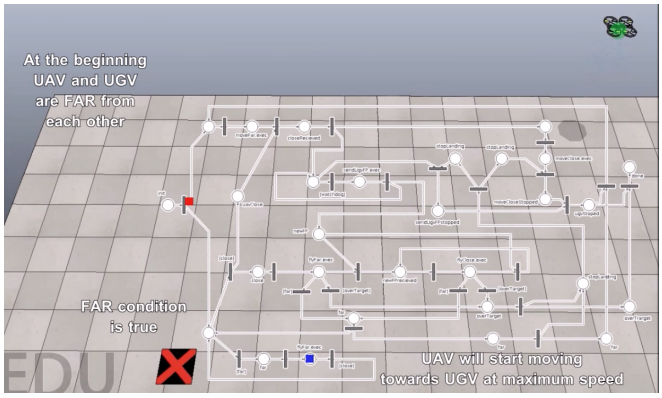
The actual positions of the UAV and UGV during the simulation are communicated to V-REP via ROS topics. The simulation environment will be updated according to the new changes. The whole system keeps on running until a final state in the Petri Net Plan is reached. Figure 2(a) shows a snapshot of the simulation when the UAV is flying toward the UGV (*flyFar* action during Phase 2).

A video showing the complete execution of the plan can be seen at the link in the footnote[3]. The vehicles start far away from each other; then the UAV flies toward the UGV with maximum speed (Phase 2) until the close condition comes true (Phase 3). At this moment, UAV sends an external event to the UGV and the UGV starts sending its

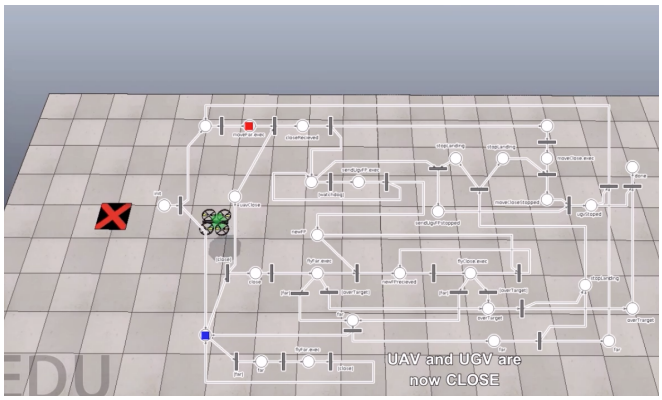
[3] `https://goo.gl/hiZIKP`



(a) Phase1: Initializing UAV and UGV in the environment



(b) Phase2: UAV flies toward UGV



(c) Phase3: UAV and UGV gets close

Figure 2. V-REP environment setup for simulating the cooperative landing task.

future positions to the UAV and decreases its speed to make the landing easier. The future position of the UGV is important for the UAV because unexpected events may happens (e.g. obstacles) that prevent the UAV to land on the UGV and so they may get far away again (from Phase 3 to Phase 2). In other words the video illustrates that the coordination between the two vehicles is not a one-step synchronization action but it is a continuous behavior. The video also shows the evolution of the simplified version of the Petri Net plan during the simulation in order to better illustrate the behavior of the system. The mission is accomplished when the UAV lands on the UGV: this corresponds to the final state (place) of the plan.

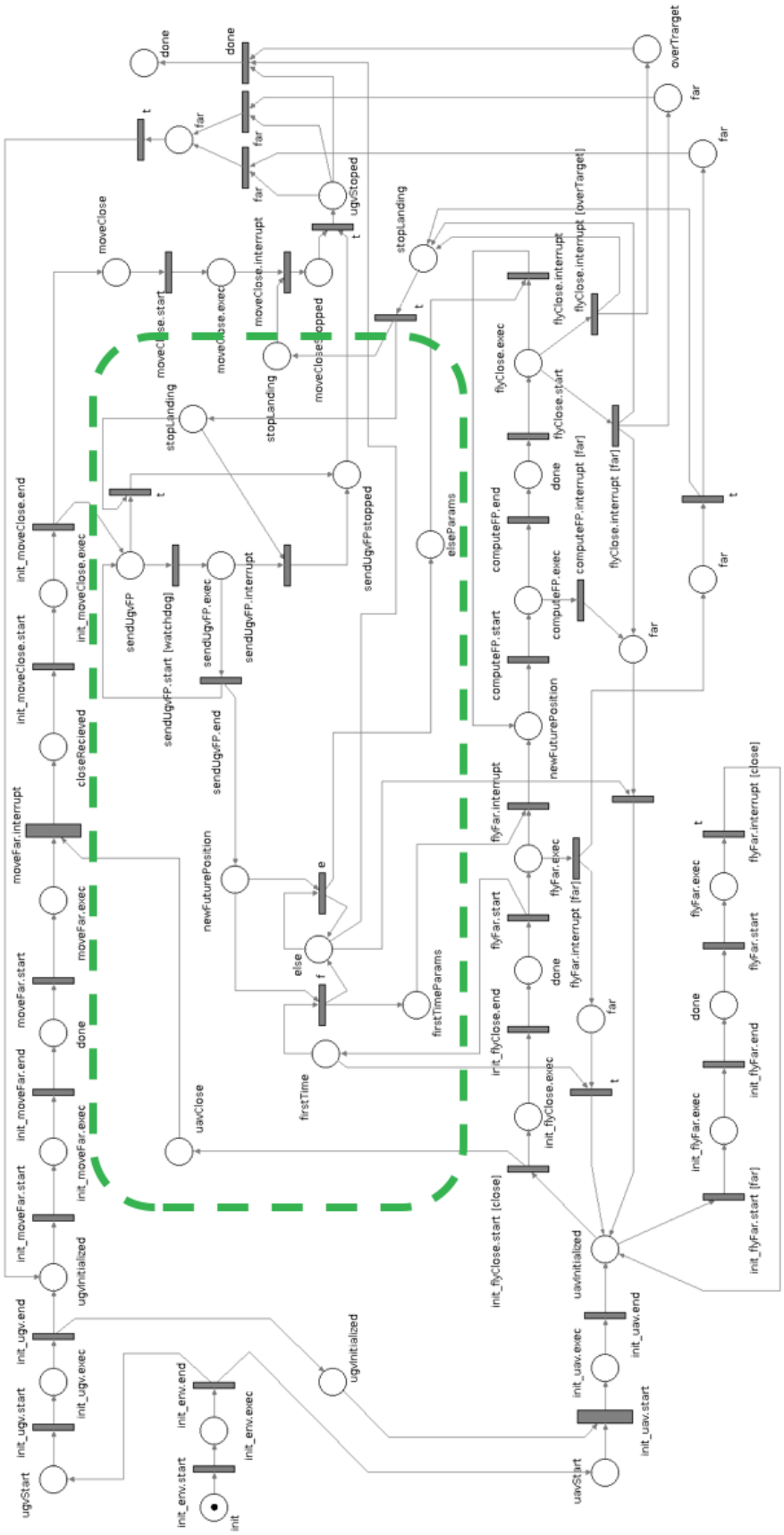


Figure 3. Petri Net Plan created by JARP editor for modeling UAV/UGV cooperative landing task. The highlighted part of the plan is specifically responsible for synchronization between UAV and UGV

6 Conclusions and Future Work

In this paper we investigate the use of a high level language or team plans to describe members actions and to monitor the behavior of vehicles during mission execution in a simulated multi robot system. In particular we considered a complex cooperative landing scenario, where an unmanned aerial vehicle (UAV) must land on an unmanned ground vehicle (UGV), and we used Petri Net Plans (PNPs) framework to model this system. The PNPs framework provides different structures for handling interruption and synchronization behavior which makes modeling and monitoring of the plans easier specifically when tight coordination among team members exist. The simulation results confirm the benefit of using a high level specification language for modeling and monitoring cooperative behavior in multi robot systems to achieve the collective behaviors even in face of unexpected events which can be recommended to many other similar applications in this area.

REFERENCES

[1] Bernard Berthomieu, Didier Lime, Olivier H Roux, and François Vernadat, 'Reachability problems and abstract state spaces for time petri nets with stopwatches', *Discrete Event Dynamic Systems*, **17**(2), 133–158, (2007).

[2] A. S. Brandao, J. A. Sarapura, E. M. d. O. Caldeira, M. Sarcinelli-Filho, and R. Carelli, 'Decentralized control of a formation involving a miniature helicopter and a team of ground robots based on artificial vision', in *Robotics Symposium and Intelligent Robotic Meeting (LARS), 2010 Latin American*, pp. 126–131, (Oct 2010).

[3] L. Cantelli, M. Mangiameli, C. D. Melita, and G. Muscato, 'Uav/ugv cooperation for surveying operations in humanitarian demining', in *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 1–6, (Oct 2013).

[4] Philip R Cohen and Hector J Levesque, 'Teamwork', *Special Issue in cognitive Science and Artificial Intelligence*, 487–512, (1991).

[5] J. M. Daly, Y. Ma, and S. L. Waslander, 'Coordinated landing of a quadrotor on a skid-steered ground vehicle in the presence of time delays', in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4961–4966, (Sept 2011).

[6] A. Dewan, A. Mahendran, N. Soni, and K. M. Krishna, 'Heterogeneous ugv-mav exploration using integer programming', in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5742–5749, (Nov 2013).

[7] Alessandro Farinelli, Nicoló Marchi, Masoume M. Raeissi, Nathan Brooks, and Paul Scerri, 'A mechanism for smoothly handling human interrupts in team oriented plans', in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '15, pp. 377–385, Richland, SC, (2015). International Foundation for Autonomous Agents and Multiagent Systems.

[8] M. K. Habib, Y. Baudoin, and F. Nagata, 'Robotics for rescue and risky intervention', in *IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society*, pp. 3305–3310, (Nov 2011).

[9] Kurt Jensen, 'Coloured petri nets', *Petri nets: central models and their properties*, 248–299, (1987).

[10] Kurt Jensen and Lars M. Kristensen, *Coloured Petri nets: Modelling and Validation of Concurrent Systems*, Springer, 2009.

[11] Gal A. Kaminka and Inna Frenkel, 'Flexible teamwork in behavior-based robots', in *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, pp. 108–113, (2005).

[12] Gal A. Kaminka and Inna Frenkel, 'Flexible teamwork in behavior-based robots', in *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, pp. 108–113, (2005).

[13] Torsten Kröger, 'On-line trajectory generation: Nonconstant motion constraints', in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 2048–2054. IEEE, (2012).

[14] Torsten Kröger and Jose Padial, 'Simple and robust visual servo control of robot arms using an on-line trajectory generator', in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 4862–4869. IEEE, (2012).

[15] Simon Lacroix and Guy Besnerais, *Robotics Research: The 13th International Symposium ISRR*, chapter Issues in Cooperative Air/Ground Robotic Systems, 421–432, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[16] Tadao Murata, 'Petri nets: Properties, analysis and applications', *Proceedings of the IEEE*, **77**(4), 541–580, (1989).

[17] M. Owen, H. Yu, T. McLain, and R. Beard, 'Moving ground target tracking in urban terrain using air/ground vehicles', in *2010 IEEE Globecom Workshops*, pp. 1816–1820, (Dec 2010).

[18] James Lyle Peterson, 'Petri net theory and the modeling of systems.', *PRENTICE-HALL, INC., ENGLEWOOD CLIFFS, NJ 07632, 1981, 290*, (1981).

[19] C. Phan and H. H. T. Liu, 'A cooperative uav/ugv platform for wildfire detection and fighting', in *System Simulation and Scientific Computing, 2008. ICSC 2008. Asia Simulation Conference - 7th International Conference on*, pp. 494–498, (Oct 2008).

[20] Milind Tambe, 'Towards flexible teamwork', *Journal of Artificial Intelligence Research*, 83–124, (1997).

[21] H. Yu, R. W. Beard, M. Argyle, and C. Chamberlain, 'Probabilistic path planning for cooperative target tracking using aerial and ground vehicles', in *Proceedings of the 2011 American Control Conference*, pp. 4673–4678, (June 2011).

[22] Erica Zawodny MacArthur, Donald MacArthur, and Carl Crane. Use of cooperative unmanned air and ground vehicles for detection and disposal of mines, 2005.

[23] V.A. Ziparo, L. Iocchi, PedroU. Lima, D. Nardi, and P.F. Palamara, 'Petri net plans', *Autonomous Agents and Multi-Agent Systems*, **23**(3), 344–383, (2011).